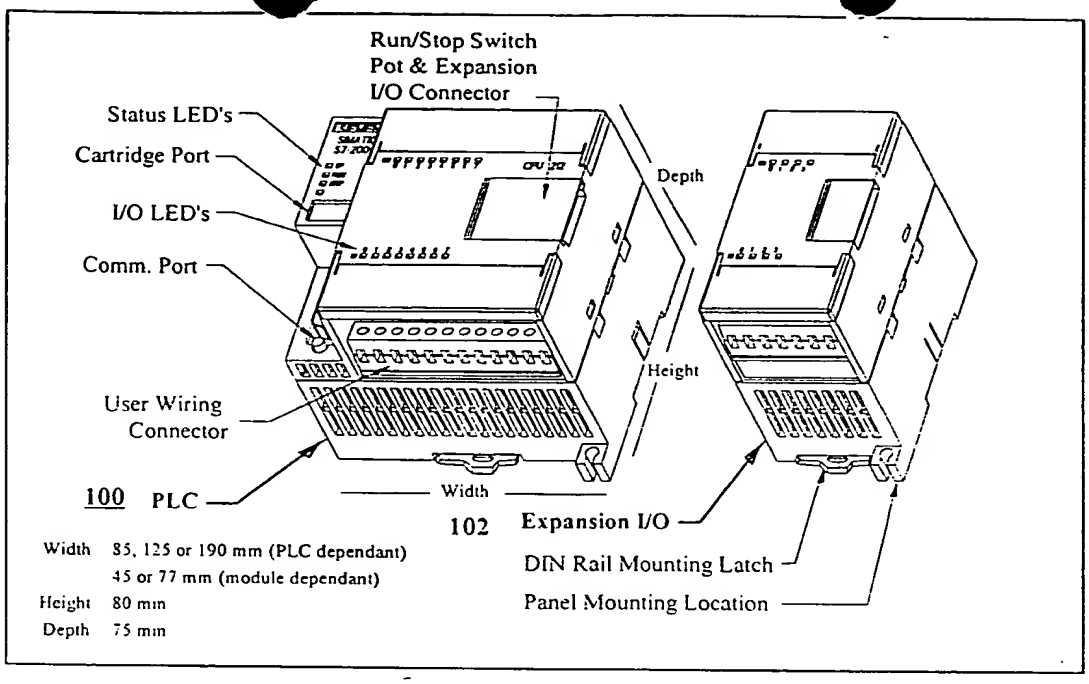


Figure 1

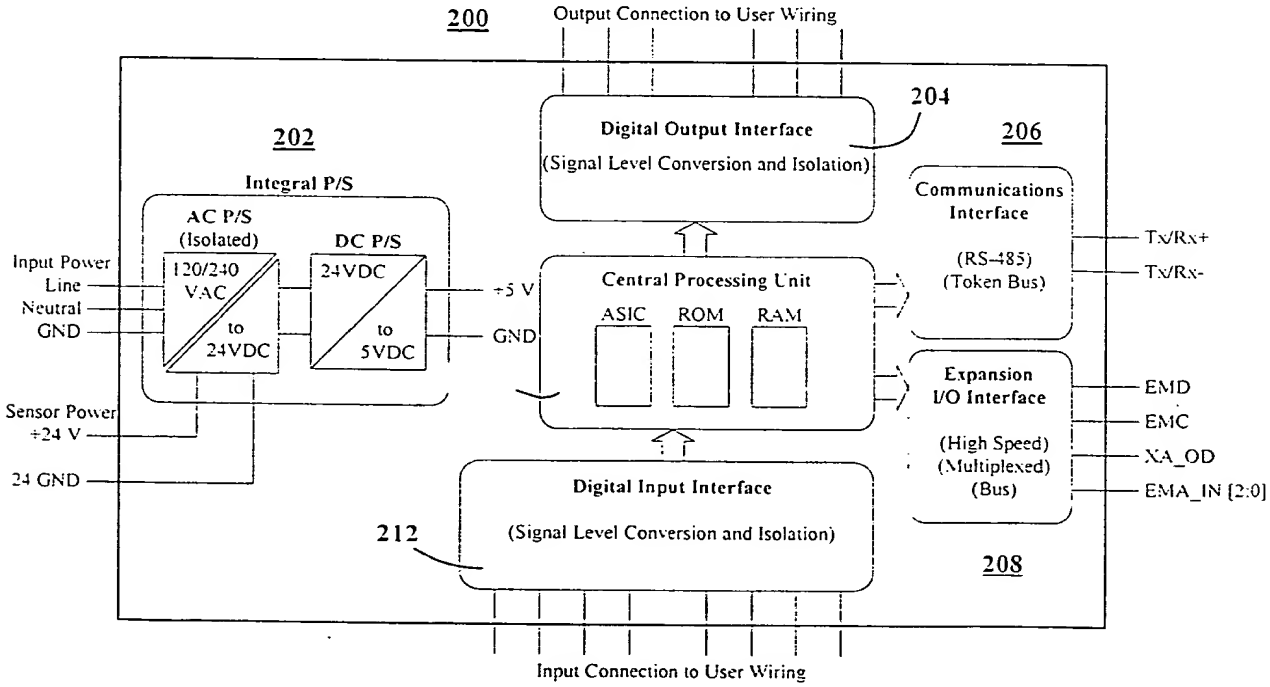


Figure 2

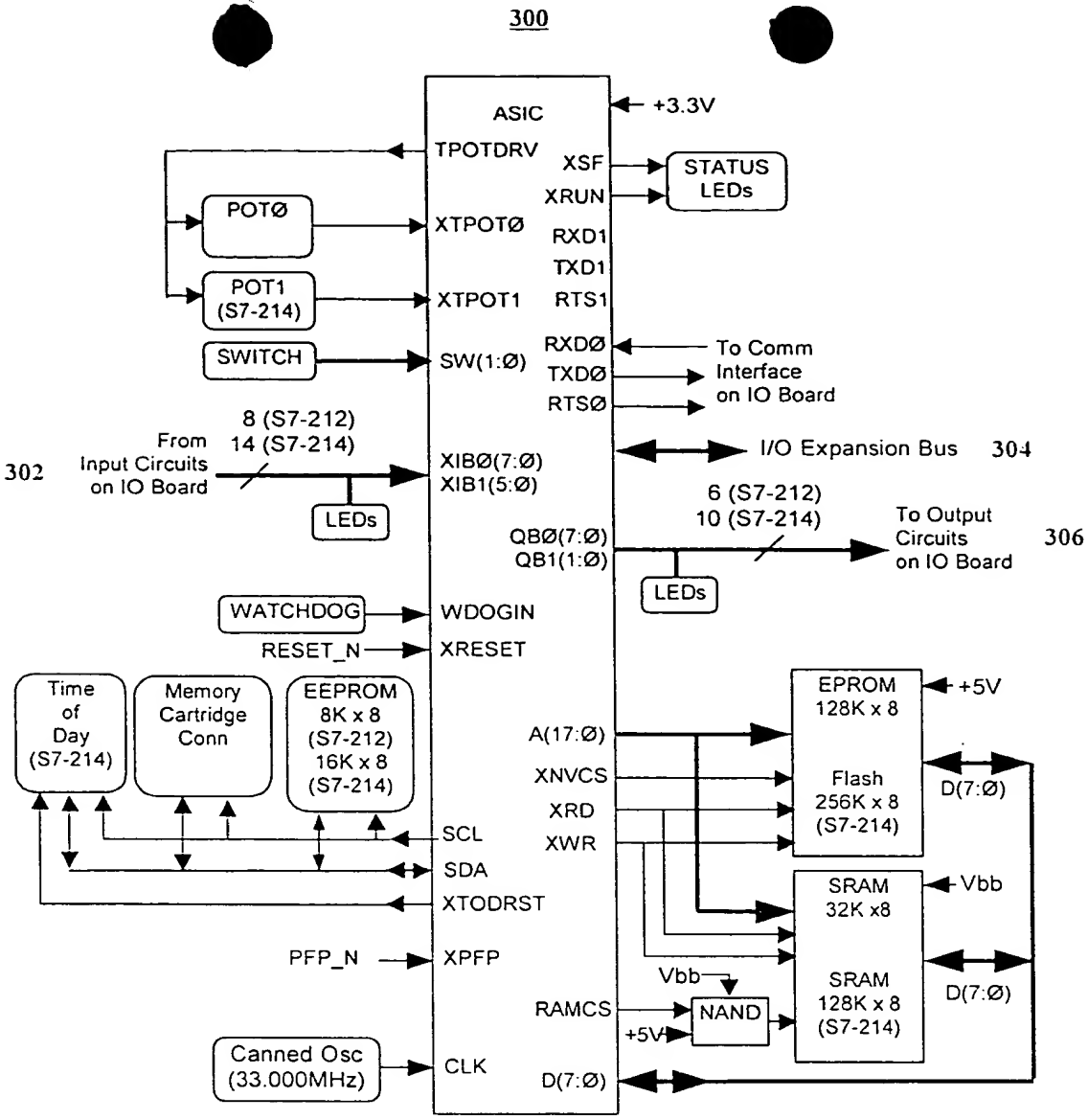

300
ASIC
+3.3V
TPOTDRV
XSF
STATUS LEDs
XRUN
POTØ
XTPOTØ
RXD1
TXD1
POT1 (S7-214)
XTPOT1
RTS1
SWITCH
SW(1:Ø)
RXDØ
To Comm Interface on IO Board
TXDØ
RTSØ
8 (S7-212)
14 (S7-214)
From Input Circuits on IO Board
302
XIBØ(7:Ø)
XIB1(5:Ø)
I/O Expansion Bus
304
LEDs
6 (S7-212)
10 (S7-214)
QBØ(7:Ø)
QB1(1:Ø)
To Output Circuits on IO Board
306
LEDs
WATCHDOG
WDOGIN
RESET_N
XRESET
Time of Day (S7-214)
Memory Cartridge Conn
EEPROM 8K x 8 (S7-212) 16K x 8 (S7-214)
A(17:Ø)
EPROM 128K x 8
+5V
XNVCS
XRD
Flash 256K x 8 (S7-214)
D(7:Ø)
XWR
SCL
SDA
XTODRST
SRAM 32K x8
Vbb
PFP_N
XPFP
Vbb
SRAM 128K x 8 (S7-214)
D(7:Ø)
RAMCS
NAND
+5V
Canned Osc (33.000MHz)
CLK
D(7:Ø)

Figure 3

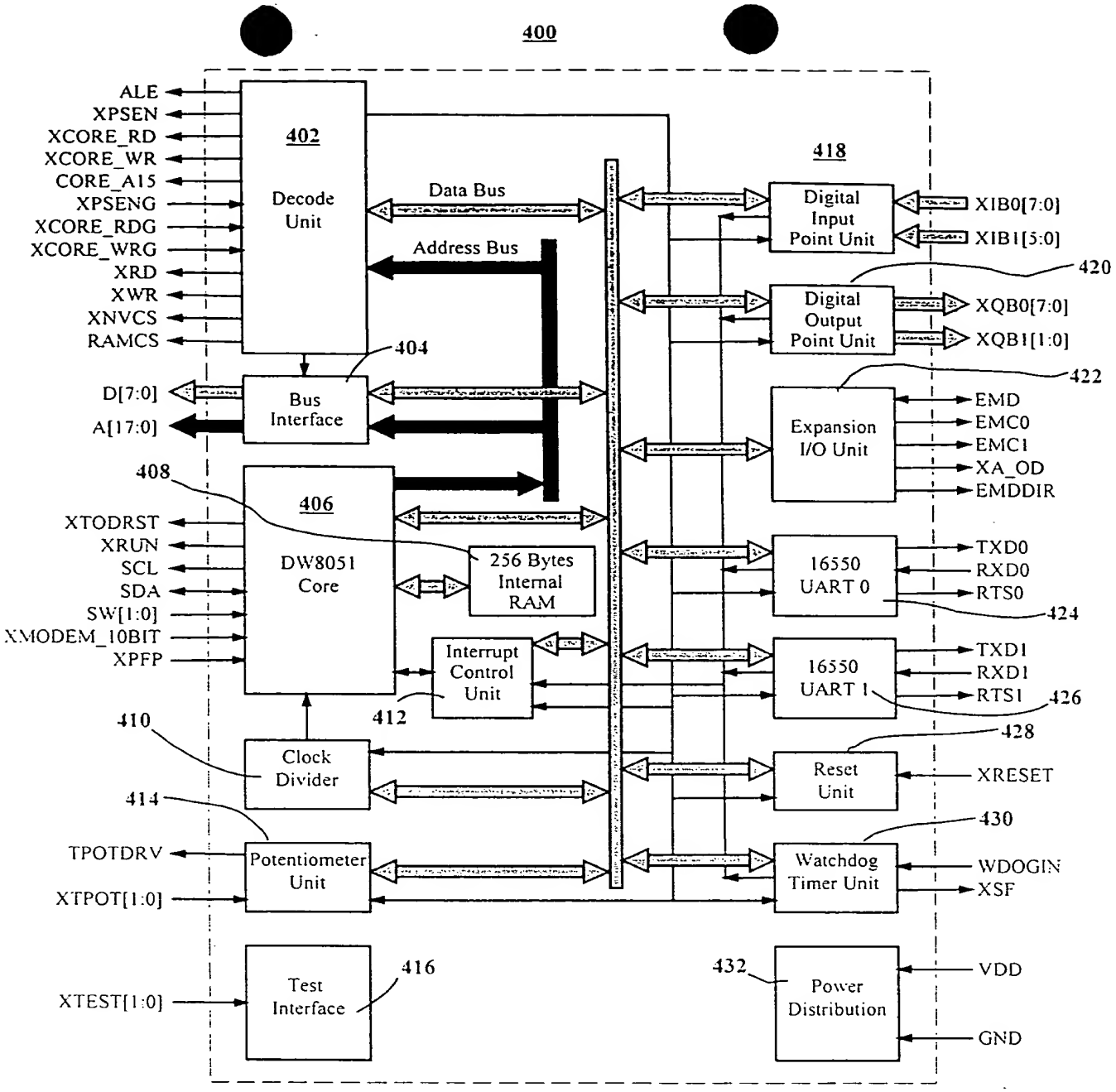

400
ALE
XPSEN
XCORE_RD
XCORE_WR
CORE_A15
XPSENG
XCORE_RDG
XCORE_WRG
XRD
XWR
XNVCS
RAMCS
402
Decode Unit
Data Bus
Address Bus
418
Digital Input Point Unit
XIB0[7:0]
XIB1[5:0]
420
Digital Output Point Unit
XQB0[7:0]
XQB1[1:0]
404
D[7:0]
A[17:0]
Bus Interface
422
Expansion I/O Unit
EMD
EMC0
EMC1
XA_OD
EMDDIR
408
406
XTODRST
XRUN
SCL
SDA
SW[1:0]
XMODEM_10BIT
XPFP
DW8051 Core
256 Bytes Internal RAM
16550 UART 0
TXD0
RXD0
RTS0
424
Interrupt Control Unit
412
16550 UART 1
TXD1
RXD1
RTS1
426
410
Clock Divider
428
Reset Unit
XRESET
414
TPOTDRV
XTPOT[1:0]
Potentiometer Unit
430
Watchdog Timer Unit
WDOGIN
XSF
XTEST[1:0]
Test Interface
416
432
Power Distribution
VDD
GND

Figure 4

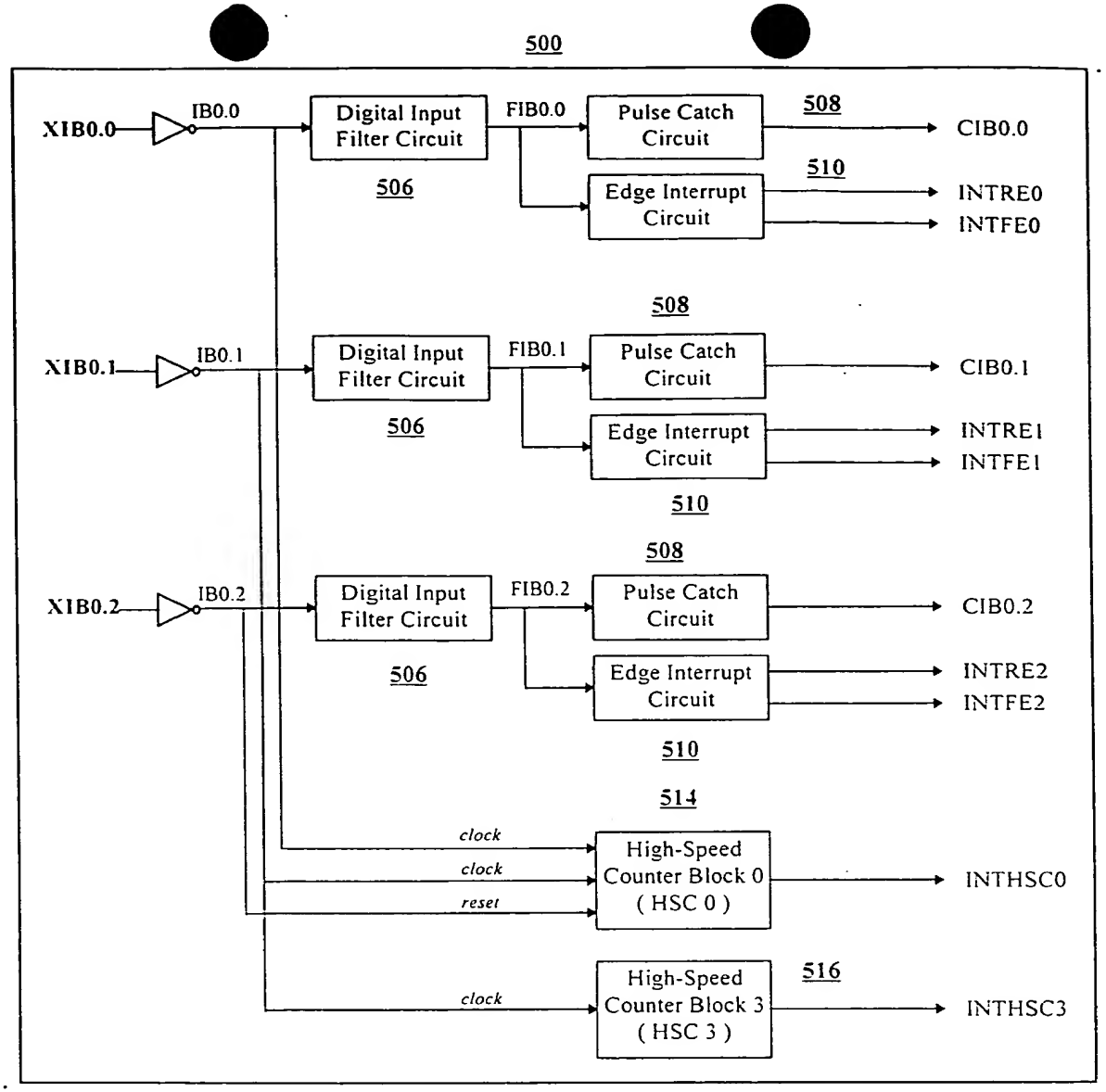

500
XIB0.0
IB0.0
Digital Input Filter Circuit
506
FIB0.0
Pulse Catch Circuit
508
CIB0.0
Edge Interrupt Circuit
510
INTRE0
INTFE0
XIB0.1
IB0.1
Digital Input Filter Circuit
506
FIB0.1
Pulse Catch Circuit
508
CIB0.1
Edge Interrupt Circuit
510
INTRE1
INTFE1
XIB0.2
IB0.2
Digital Input Filter Circuit
506
FIB0.2
Pulse Catch Circuit
508
CIB0.2
Edge Interrupt Circuit
510
INTRE2
INTFE2
514
clock
clock
reset
High-Speed Counter Block 0 (HSC 0)
INTHSC0
clock
High-Speed Counter Block 3 (HSC 3)
516
INTHSC3

Figure 5

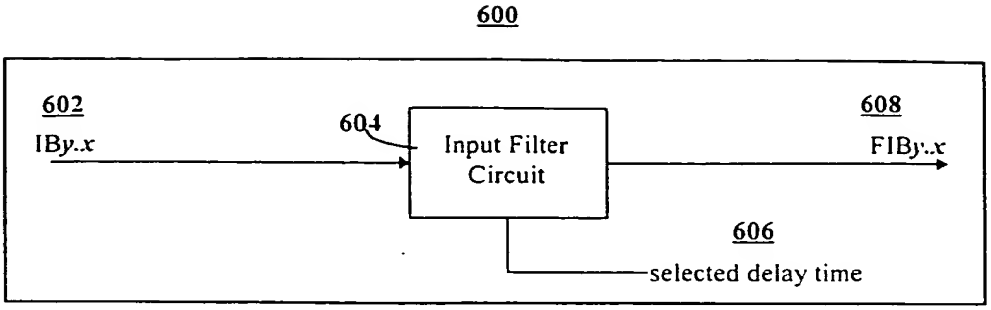


Figure 6a

Digital Filter Operation Definition

Input Point State	Current Count	Next Count	Present Output Value	Next Output Value
0 (decrements counter)	0	0	0	0
	n, where 3 ≥ n > 0	n - 1	0	0
	4	3	1	0
	n, where 15 ≥ n > 4	n - 1	1	1
1 (increments counter)	n, where 11 > n ≥ 0	n + 1	0	0
	11	12	0	1
	n, where 14 ≥ n > 11	n + 1	1	1
	15	15	1	1

Figure 6b

604

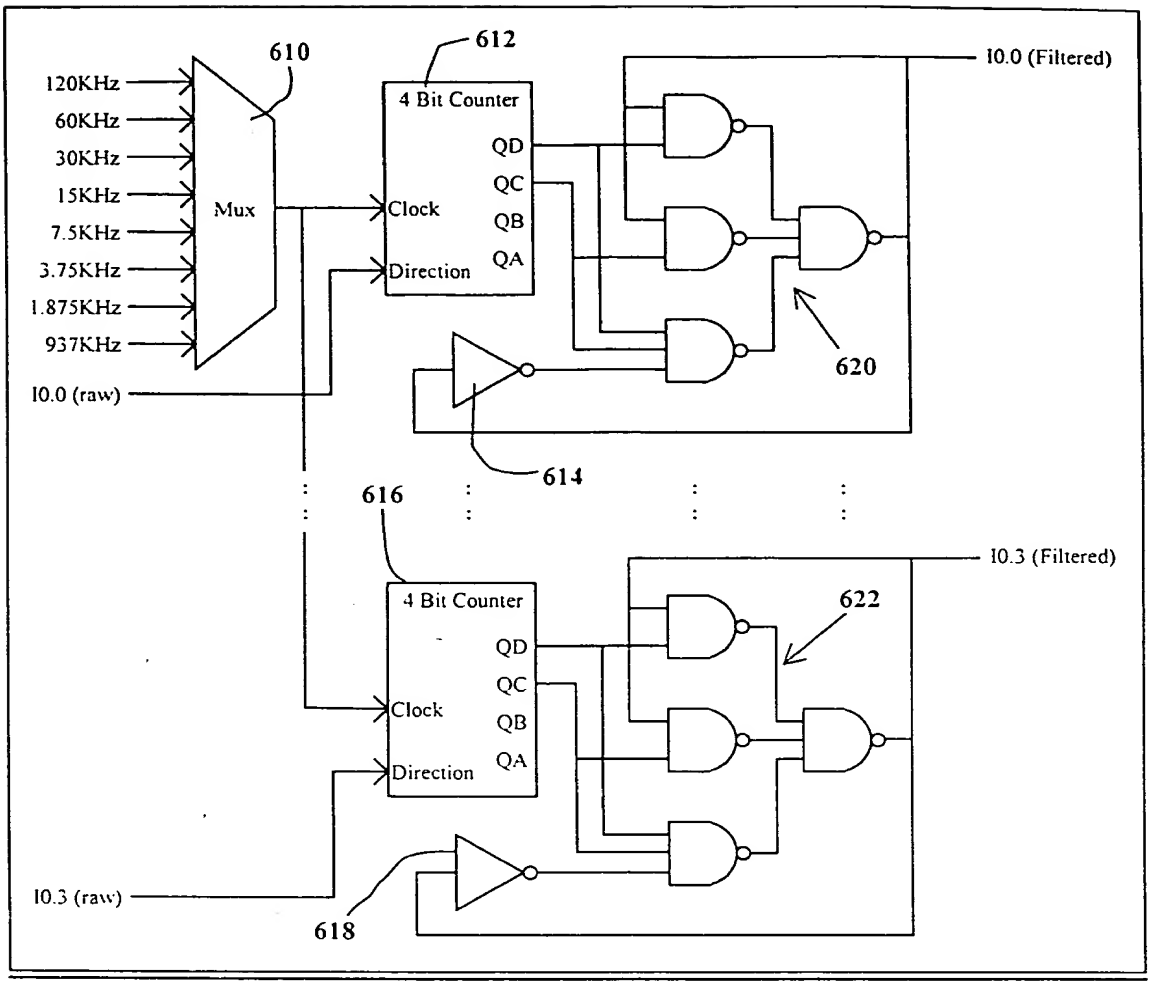


Figure 6c

Frequency	Period	Number of Counts	Delay time
120 KHz	8.33 μsec	12	0.1 ms
60 KHz	16.6 μsec	12	0.2 ms
30 KHz	33.3 μsec	12	0.4 ms
15 KHz	66.7 μsec	12	0.8 ms
7.5 KHz	133 μsec	12	1.6 ms
3.75 KHz	267 μsec	12	3.2 ms
1.875 KHz	533 μsec	12	6.4 ms
937 KHz	1067 μsec	12	12.8 ms

Figure 6d

Register Value	Corresponding delay time
00	0.2 ms
01	0.4 ms
02	0.8 ms
03	1.6 ms [1]
04	1.6 ms [1]
05	3.2 ms
06	6.4 ms
07	12.8 ms
08 to FF	no delay

[1] Two selections for 1.6 ms delay time exist for 1st generation ASIC compatibility reasons

Figure 7

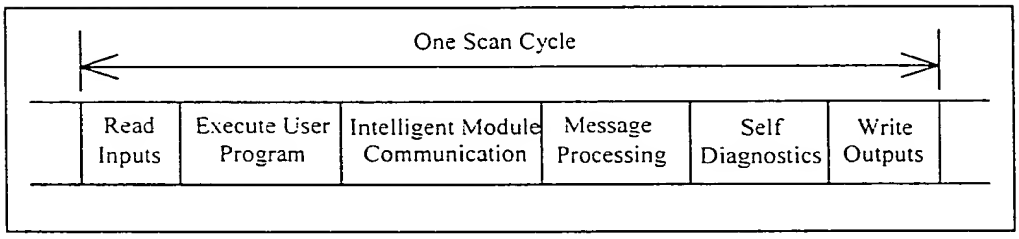


Figure 8

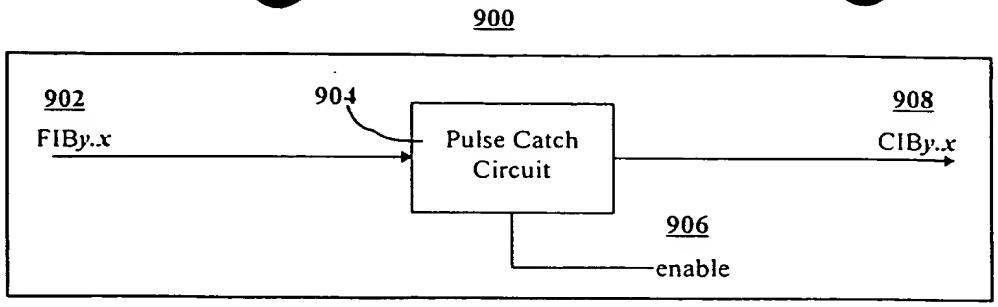


Figure 9

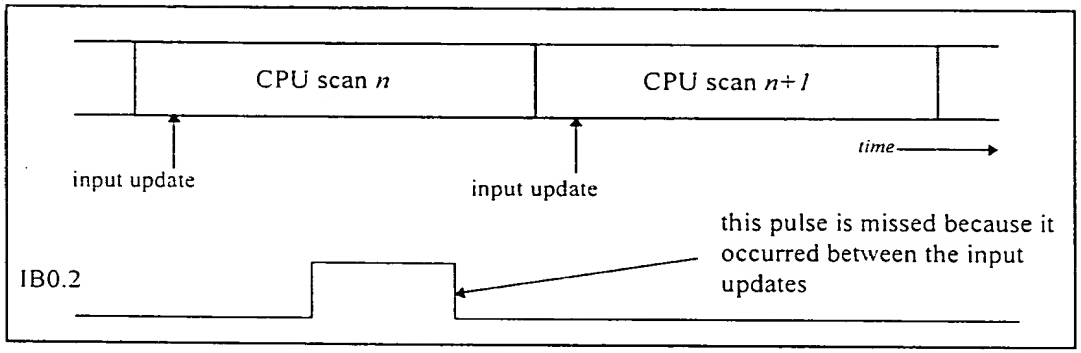


Figure 10

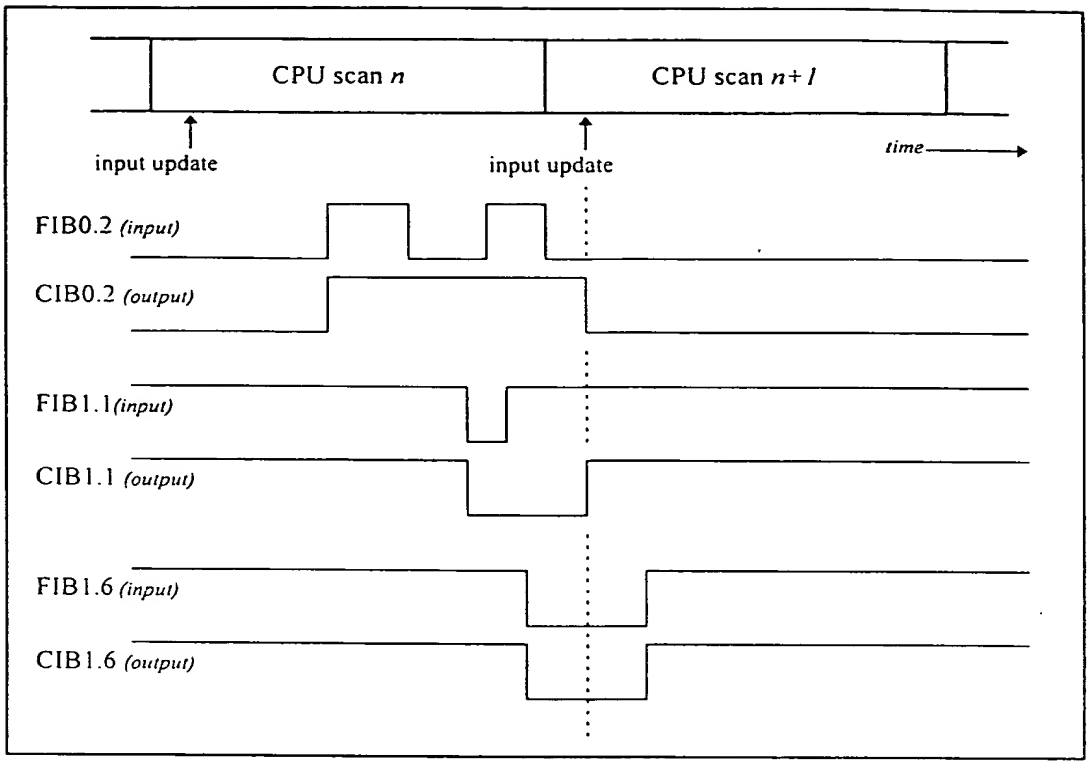

CPU scan n
CPU scan n+1
input update
input update
time
FIB0.2 (input)
CIB0.2 (output)
FIB1.1 (input)
CIB1.1 (output)
FIB1.6 (input)
CIB1.6 (output)

Figure 11

PCE	PS CV	PS F	RP	NS CV	NS F	Comment
1	I	0	-	CV	0	I = CV and F is not set; RP is a don't care
1	not I	0	-	I	1	I ≠ CV; capture new value of I and set F = 1
1	-	1	0	CV	1	CV has not been read
1	-	1	1	I	0	CV has been read; set CV = I
0	-	-	-	I	0	Pulse catch disabled; CV = I

Figure 12a

Pulse Catch Enable Registers

Address	Description
0002H	register name: **IB0_Pulse_Catch_Enable_Register (IB0PCE)** size: **byte (8-bit)** access: **read / write** reset value: **00H** 7 … 0 EN7 | EN6 | EN5 | EN4 | EN3 | EN2 | EN1 | EN0 ENx: 1 = enables pulse catch operation on input point IB0.x 0 = disables pulse catch operation on input point IB0.x
0003H	register name: **IB1_Pulse_Catch_Enable_Register (IB1PCE)** size: **byte (8-bit)** access: **read / write** reset value: **00H** 7 … 0 x | x | EN5 | EN4 | EN3 | EN2 | EN1 | EN0 ENx: 1 = enables pulse catch operation on input point IB1.x 0 = disables pulse catch operation on input point IB1.x

Figure 12b

register **IB0PS:**	Read of this register returns **CIB0[7:0]** and retriggers pulse catch circuits for IB0 input points	used by SW for input update
register **IB1PS:**	Read of this register returns **CIB1[5:0]** and retriggers pulse catch circuits for IB1 input points	used by SW for input update
register **IB0PSNR:**	Read of this register returns **CIB0[7:0]** and leaves pulse catch circuits unaffected	used by SW for immediate access
register **IB1PSNR:**	Read of this register returns **CIB1[5:0]** and leaves pulse catch circuits unaffected	used by SW for immediate access

Figure 12c

Input Point Status Registers	
Address	Description
0004H	register name: IB0_Input_Point_Status_Register (IB0PS) size: byte (8-bit) access: read only reset value: 00H 7 … 0 CI7 | CI6 | CI5 | CI4 | CI3 | CI2 | CI1 | CI0 CIx: Contains conditioned input point state CIB0.x
0005H	register name: IB1_Input_Point_Status_Register (IB1PS) size: byte (8-bit) access: read only reset value: 00H 7 … 0 x | x | CI5 | CI4 | CI3 | CI2 | CI1 | CI0 CIx: Contains conditioned input point state CIB1.x.
0006H	register name: IB0_Input_Point_Status_Register_No_Retrigger (IB0PSNR) size: byte (8-bit) access: read only reset value: 00H 7 … 0 CI7 | CI6 | CI5 | CI4 | CI3 | CI2 | CI1 | CI0 CIx: Contains conditioned input point state CIB0.x
0007H	register name: IB1_Input_Point_Status_Register_No_Retrigger (IB1PSNR) size: byte (8-bit) access: read only reset value: 00H 7 … 0 x | x | CI5 | CI4 | CI3 | CI2 | CI1 | CI0 CIx: Contains conditioned input point state CIB1.x.

Figure 12d

1200e

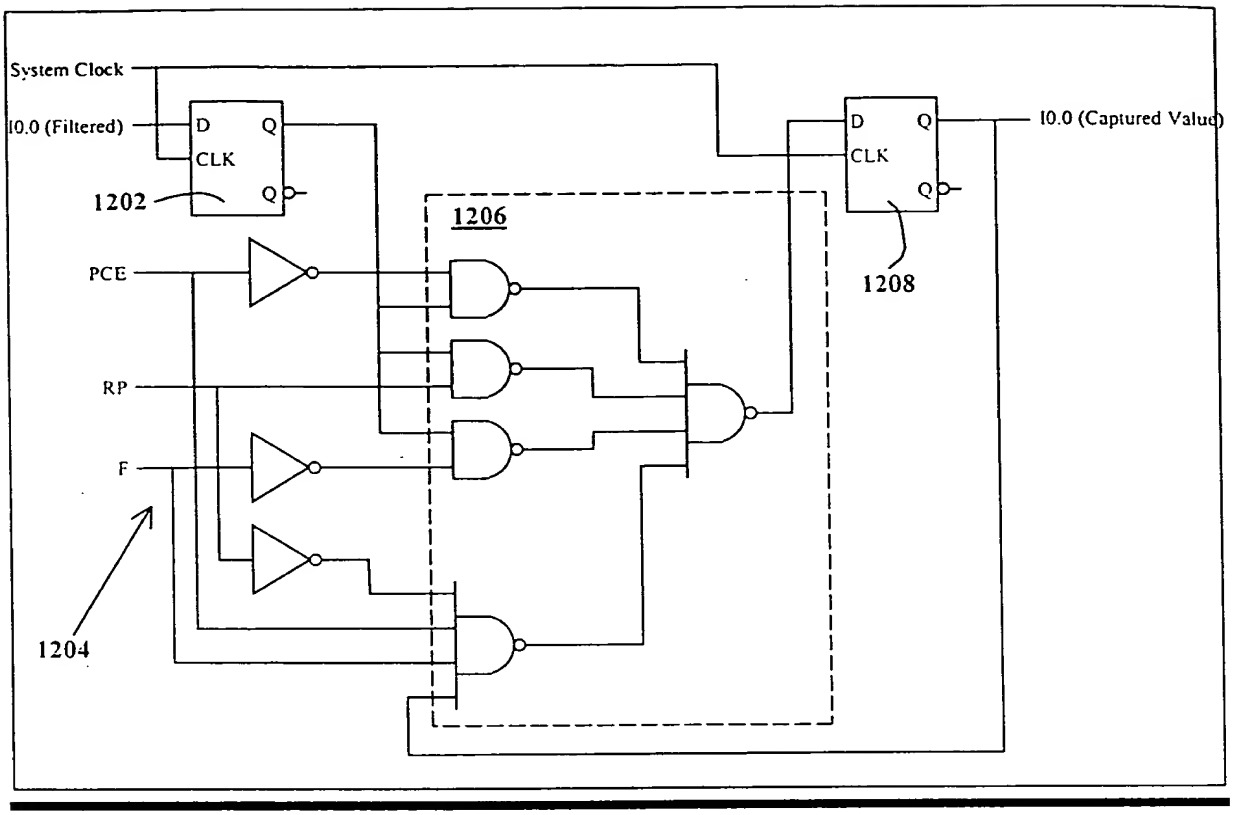


Figure 12e

1200f

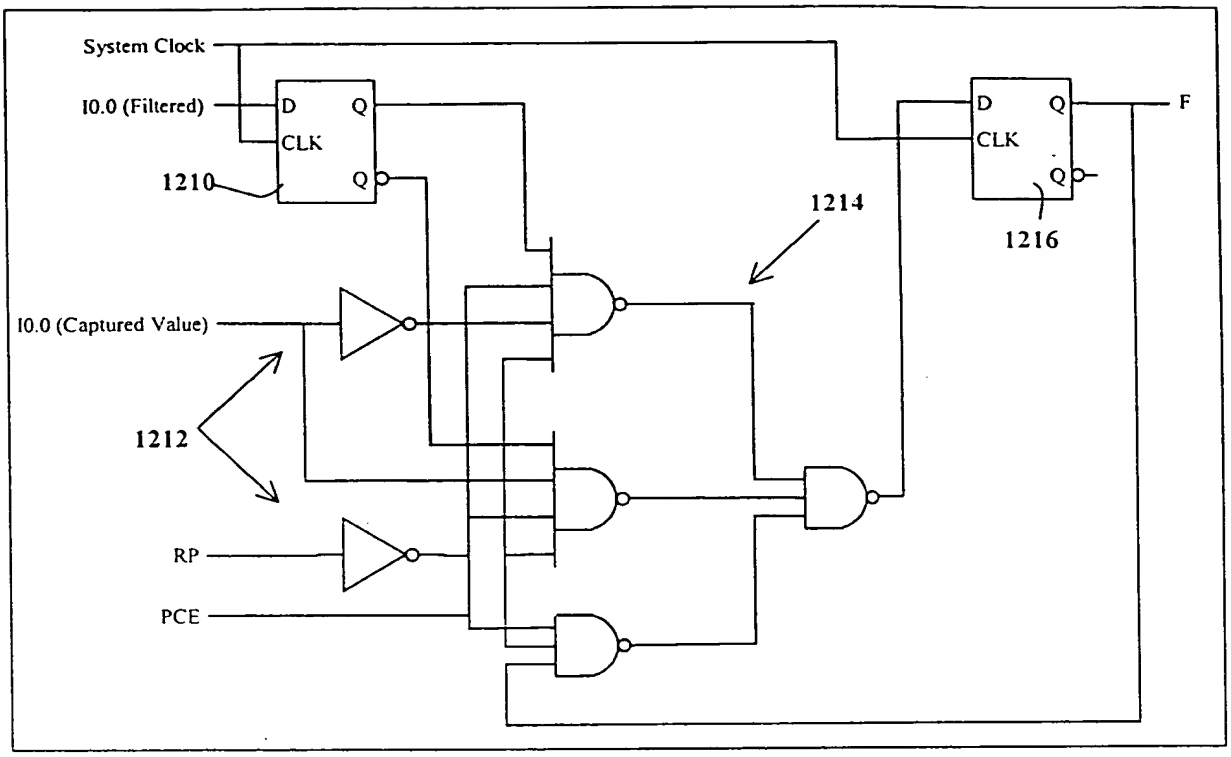


Figure 12f

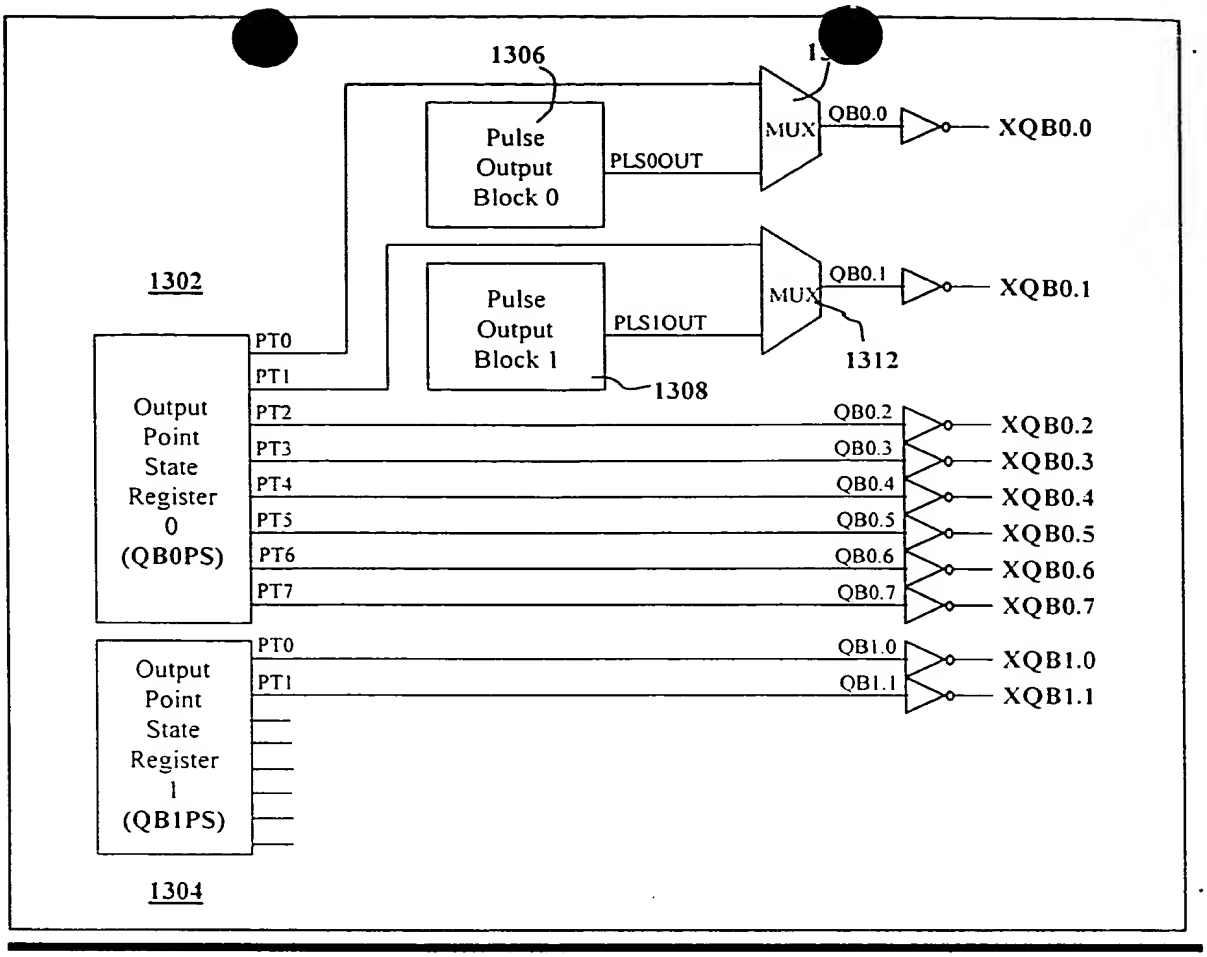

1300
1306
Pulse Output Block 0
PLS0OUT
MUX
QB0.0
XQB0.0
1302
Pulse Output Block 1
PLS1OUT
MUX
QB0.1
XQB0.1
1312
1308
PT0
PT1
Output Point State Register 0 (QB0PS)
PT2
PT3
PT4
PT5
PT6
PT7
QB0.2
QB0.3
QB0.4
QB0.5
QB0.6
QB0.7
XQB0.2
XQB0.3
XQB0.4
XQB0.5
XQB0.6
XQB0.7
PT0
PT1
Output Point State Register 1 (QB1PS)
QB1.0
QB1.1
XQB1.0
XQB1.1
1304

Figure 13

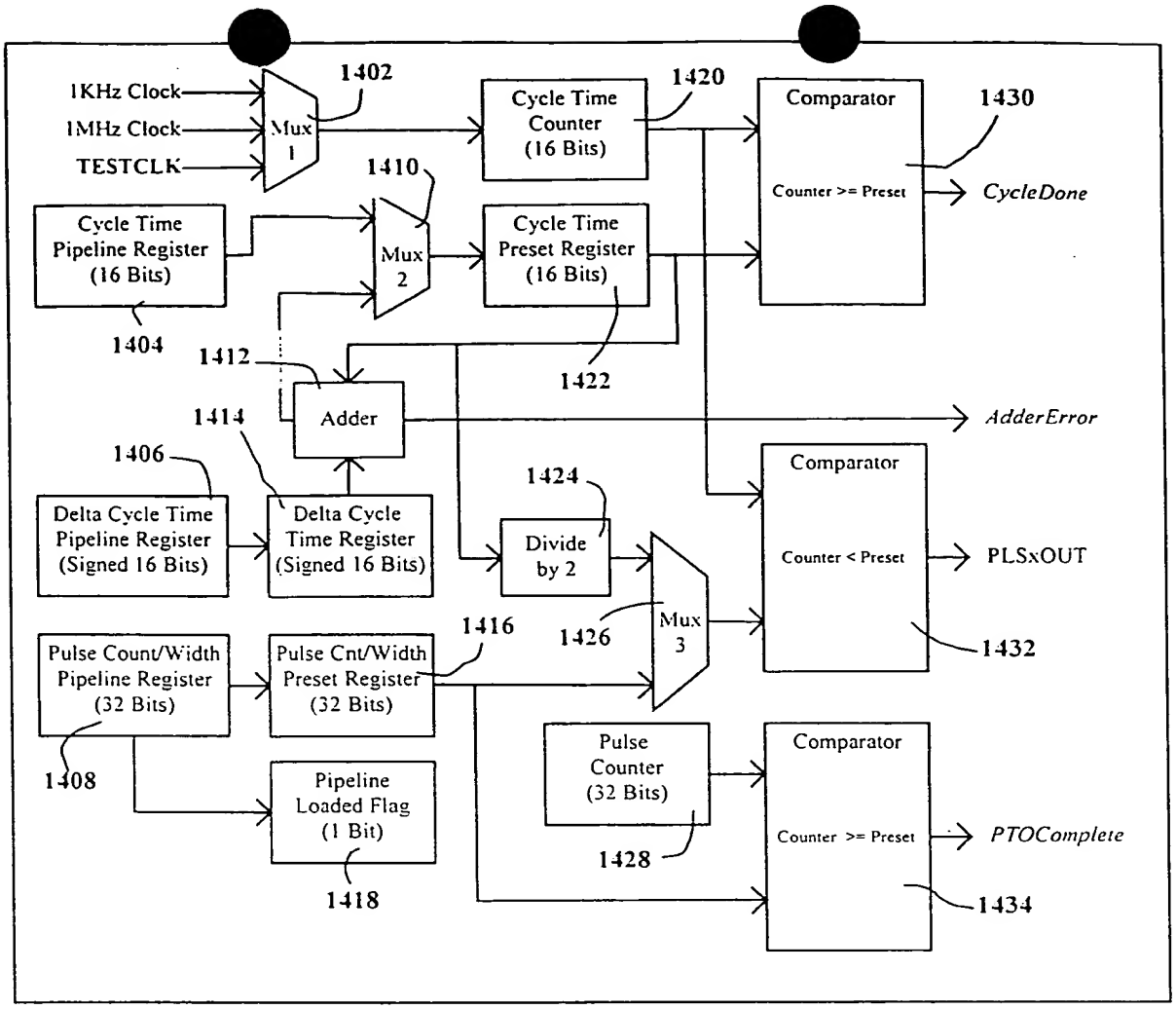

1400
1KHz Clock
1MHz Clock
TESTCLK
Mux 1
1402
Cycle Time Counter (16 Bits)
1420
Comparator
Counter >= Preset
1430
CycleDone
1410
Cycle Time Pipeline Register (16 Bits)
Mux 2
Cycle Time Preset Register (16 Bits)
1404
1412
1422
Adder
1414
AdderError
1406
Delta Cycle Time Pipeline Register (Signed 16 Bits)
Delta Cycle Time Register (Signed 16 Bits)
1424
Divide by 2
Comparator
Counter < Preset
PLSxOUT
1426
Mux 3
1432
1416
Pulse Count/Width Pipeline Register (32 Bits)
Pulse Cnt/Width Preset Register (32 Bits)
1408
Pipeline Loaded Flag (1 Bit)
1418
Pulse Counter (32 Bits)
Comparator
Counter >= Preset
PTOComplete
1428
1434

Figure 14a

Register Name	Valid Value Range
Cycle Time Preset Register Cycle Time Pipeline Register	2 to 65535
Delta Cycle Time Register Delta Cycle Pipeline Register	-32768 to 32767
Pulse Count/Width Preset Register Pulse Count/Width Pipeline Register	1 to (2^{32}-1) 0 to 65535 PTO mode PWM mode

Figure 14b

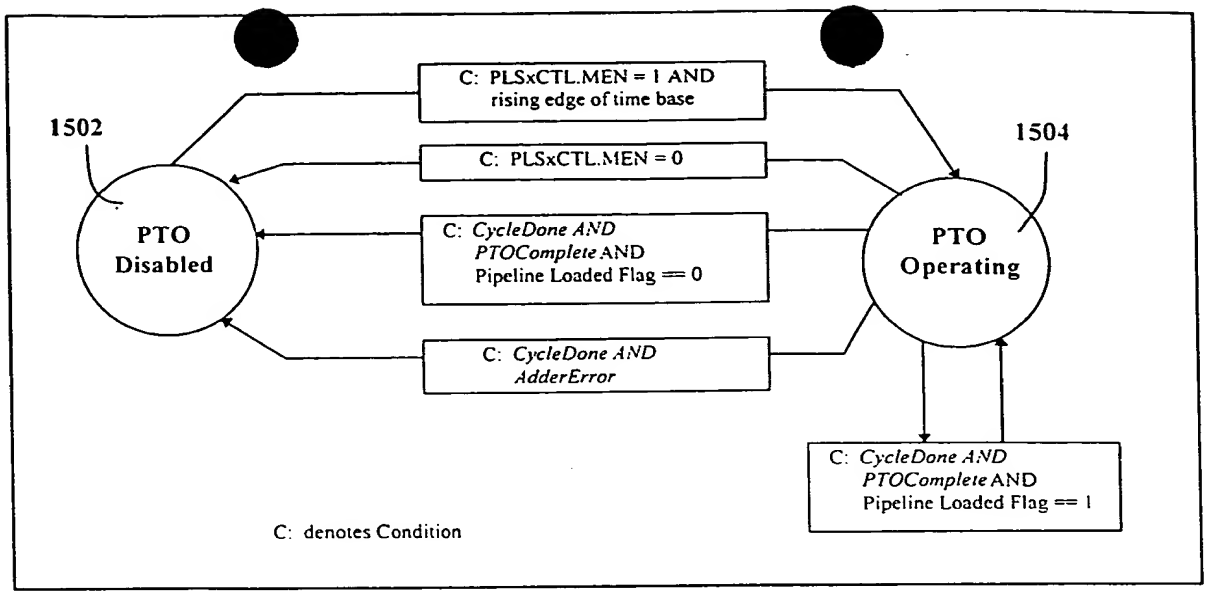

1502
1504
C: PLSxCTL.MEN = 1 AND
rising edge of time base
C: PLSxCTL.MEN = 0
C: CycleDone AND
PTOComplete AND
Pipeline Loaded Flag == 0
C: CycleDone AND
AdderError
PTO
Disabled
PTO
Operating
C: CycleDone AND
PTOComplete AND
Pipeline Loaded Flag == 1
C: denotes Condition

Figure 15

```
cycle time counter = cycle time counter + 1;
IF (cycle time counter >= cycle time preset) THEN
BEGIN  // this is the CycleDone event
       -------
       pulse counter = pulse counter + 1;
       IF (pulse counter >= pulse count preset) THEN
       BEGIN  // this is the PTOComplete event
              --------
              assert INTxPLS signal, if PTOComplete interrupts are enabled;
              IF (pipeline loaded flag is set) THEN
              BEGIN
                     -------
                     transfer values from pipeline registers into operating registers;
                     set pulse counter = 0;
                     clear pipeline loaded flag;
                     --------
              END
              ELSE   // pipeline loaded flag is not set
              BEGIN
                     -------
                     GOTO PTO Disabled state;  // disable the PLS block now
                     --------
              END
              ENDIF
              -------
       END

       ELSE   // not yet at PTOComplete
       BEGIN
              -------
              cycle time preset = cycle time preset + delta cycle time;
              IF (cycle time preset exceeds bounds) THEN
              BEGIN  // this is the AdderError event
                     --------
                     assert INTxPLS signal, if AdderError interrupts are enabled;
                     GOTO PTO Disabled state;  // disable the PLS block now
                     --------
              END
              ENDIF
              -------
       END
       ENDIF
       set cycle time counter = 0;
       --------
END
ENDIF

IF (cycle time counter >= (1/2 * cycle time preset)) THEN
       PLSxOUT = 0;
ELSE   // output still in logic high portion of the current cycle
       PLSxOUT = 1;
ENDIF
```

Figure 16

1700

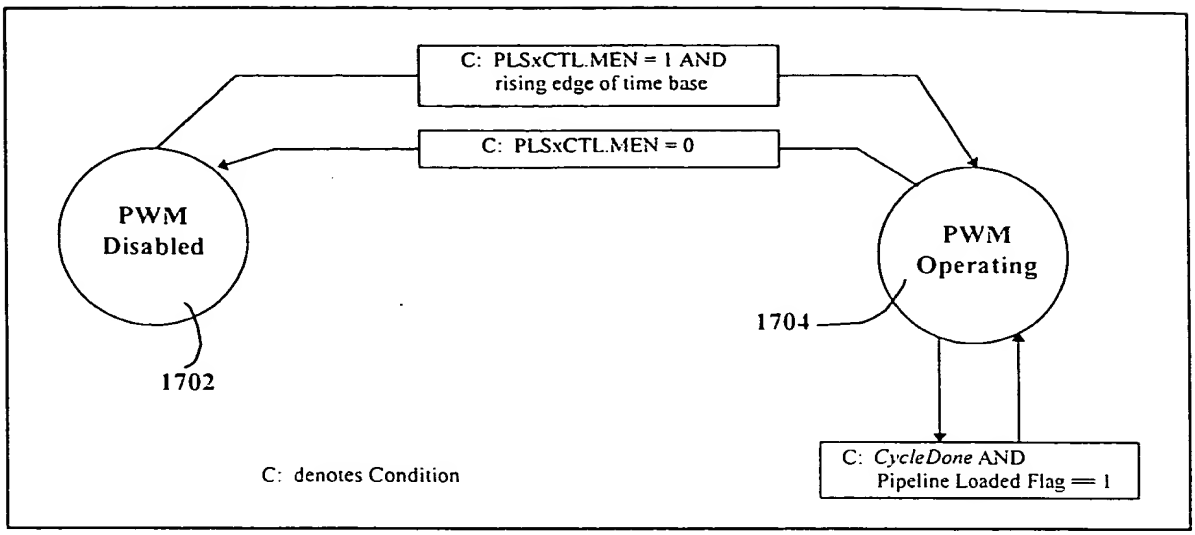


Figure 17

```
cycle time counter = cycle time counter + 1;
IF (cycle time counter >= cycle time preset) THEN
BEGIN  // this is the CycleDone event
       -------
       IF (pipeline loaded flag is set) THEN
       BEGIN
              -------
              transfer values from pipeline registers into operating registers;
              clear pipeline loaded flag;
              --------
       END
       ENDIF
       set cycle time counter = 0;
       --------
END
ENDIF

IF (cycle time counter >= (pulse width preset)) THEN
       PLSxOUT = 0;
ELSE   // output still in logic high portion of the current cycle
       PLSxOUT = 1;
ENDIF
```

Figure 18

HIGH SPEED OPERATIONS				
Instruction	Mnemonic & Operand(s)	Description	STL Status Element	VALID OPERANDS
Pulse Train Output Profile	PTOP t, n	When S0 = 1, the PTO profile specified in the table for output n is executed. ENO ← S0 * /e	STK, <current step>	Enable: S0 Table: VB, IB, QB, MB, (UI) SMB, SB, LB, *VD, *AC Output: KW (UI) 0 - 1

Definition of the TABLE for PTOP:

Byte Offset	Segment	Description of Table Entries
0		Number of profile segments (40 segments maximum)
1		Current step number being executed
2	#1	Number of steps (4 steps minimum)
4		Starting cycle time for this segment (2 to 65535 µsec)
6		Change in cycle time per step (signed value) (0 to 65535 µsec)
8	#2	Number of steps (4 steps minimum)
10		Starting cycle time for this segment (2 to 65535 µsec)
12		Change in cycle time per step (signed value) (0 to 65535 µsec)
:	:	:
:	:	:

Figure 19

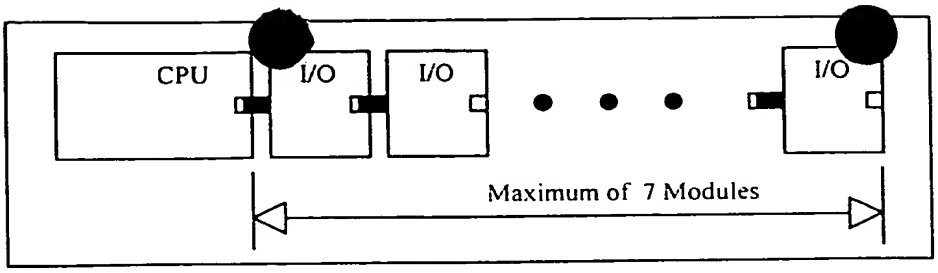


Figure 20

Signal Name	Use
EMD	Expansion Module Data - a bi-directional signal used to communicate the address and data to and from the module.
EMC[1:0]	Expansion Module Clocks - One clock is used to access expansion I/O external to the PLC, while the other is used to access I/O that is local to the PLC that does not connect directly to the ASIC.
XA_OD	Address/Output Disable - a dual function signal used to reset the state machine in the modules on the first clock of each access cycle (active low) and used to indicate output disable when a fatal error has been detected (active low for an RC time constant).
EMDDIR	Expansion Module Data Direction - this signal indicates the direction of data flow on the EMD signal line. 0 - Data is driven by the module to the PLC 1 - Data is driven by the PLC to the module
EMA[2:0]	Expansion Module Address - these signals are daisy chained from PLC to module to module. The value input to a module becomes that module's address. The module will output its address plus one to the next module. The PLC drives these signals to 0 so that the module connected to the PLC has the address of 0.
+5V	5 volt power supply - Two signals carry +5 volts.
GND	Power supply return - Two signals carry ground.

Figure 21

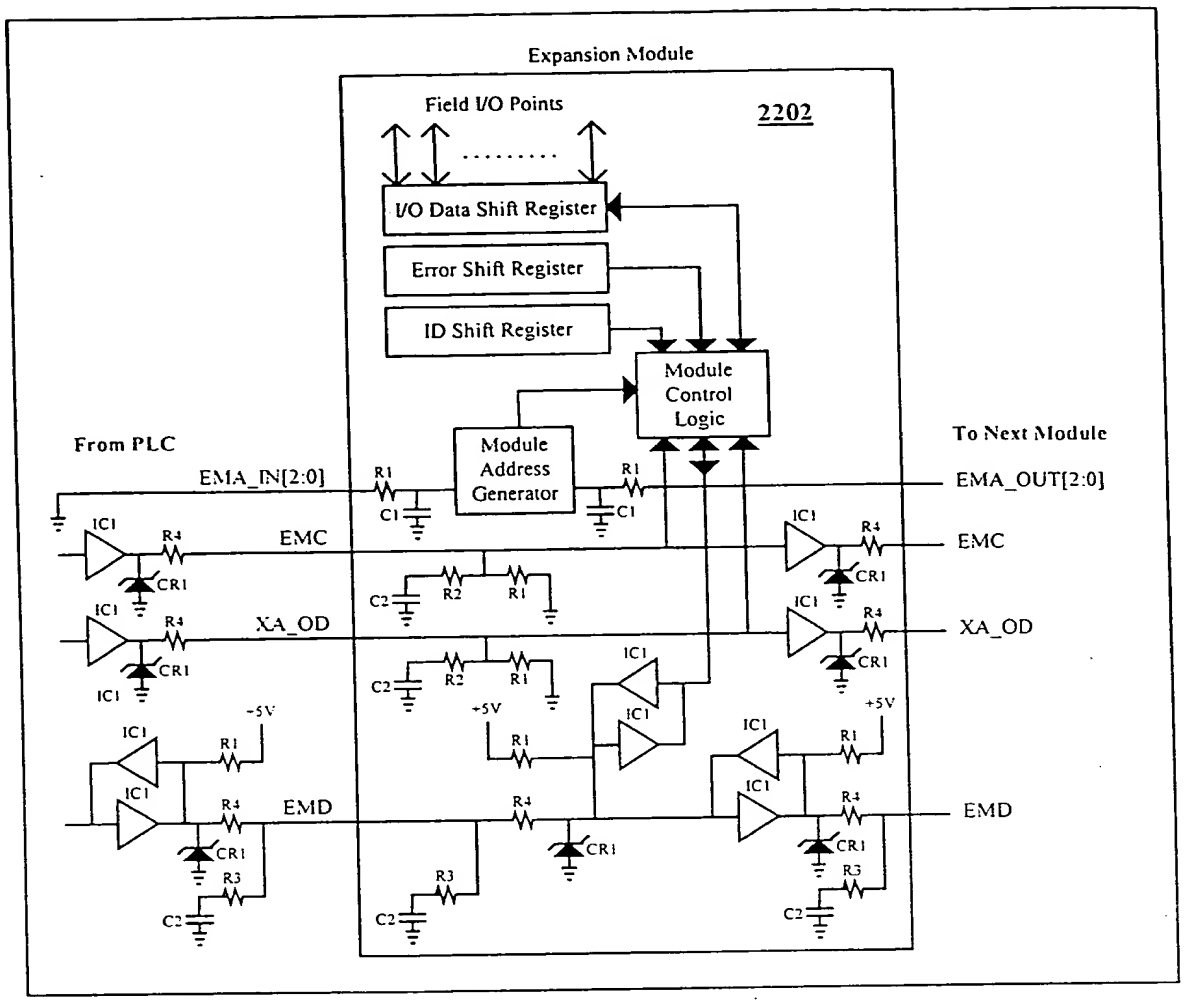

Expansion Module
Field I/O Points
2202
I/O Data Shift Register
Error Shift Register
ID Shift Register
Module Control Logic
Module Address Generator
From PLC
To Next Module
EMA_IN[2:0]
EMA_OUT[2:0]
EMC
XA_OD
EMD
IC1
R1
R2
R3
R4
C1
C2
CR1
+5V

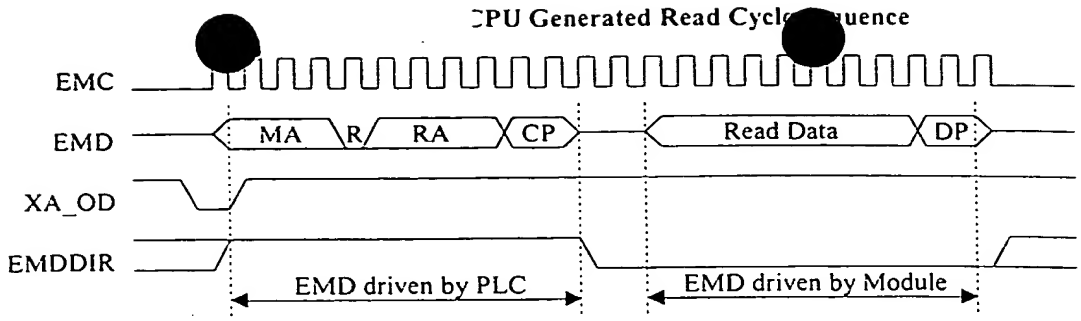
Figure 22

Reference Designator	Component Value/Type	Component Description
IC1	74ABT125/74ABT126	Tri-state buffer
CR1	TVS 5.6V Zener	Diode
R1	4.7K ohm	Resistor
R2	110 ohm	Resistor
R3	220 ohm	Resistor
R4	22 ohm	Resistor
C1	0.1µF	Capacitor
C2	100pF	Capacitor

Figure 23

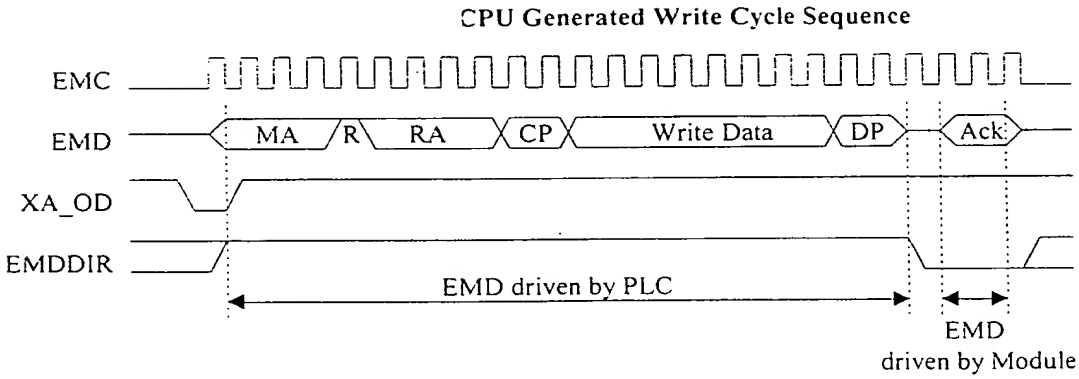
Signal Name	Driving Device	Drive Levels				Receive Levels	
		V_{OL} (Volts) (Max)	I_{OL} (mA) (Min)	V_{OH} (Volts) (Max)	I_{OH} (mA) (Min)	V_{IL} (Volts) (Max)	V_{IH} (Volts) (Min)
EMA[2:0]	Module	0.5	3.0	2.4	-3.0	0.8	2.0
XA_OD	CPU	0.55	64.0	2.0	-32.0	0.8	2.0
EMC	CPU	0.55	64.0	2.0	-32.0	0.8	2.0
EMD	CPU	0.55	64.0	2.0	-32.0	0.8	2.0
	Module	0.55	64.0	2.0	-32.0	0.8	2.0

Figure 24



MA - Module Address for modules 0 to 6 (3 bits)
R - Read/Write bit (Read active low)
RA - Register Address for registers R0 to R15 (4 bits)
CP - Control Parity generated by the CPU on MA, R, and RA (2 bits)
Read Data - Data read from the Module (8 bits)
DP - Data Parity generated by the module on read data (2 bits)

Figure 25



MA - Module Address for modules 0 to 6 (3 bits)
R - Read/Write bit (Read active low)
RA - Register Address for registers R0 to R15 (4 bits)
CP - Control Parity generated by the CPU on MA, R, and RA (2 bits)
Write Data - Data written to the Module (8 bits)
DP - Data Parity generated by the CPU on write data (2 bits)
Ack - Module acknowledge of a good write cycle (2 bits)

Figure 26

	a	*b*	*c*	*d*	
[redacted]	bit 0	bit 2	bit 4	bit 6	[redacted]
P1	bit 1	bit 2	bit 3	bit 5	bit 7
5 bit odd parity = ! (((*a* xor *b*) xor (*c* xor *d*)) xor *e*)					

Figure 27

PLC Type	Port 0	Port 1
CPU 212	256	-
CPU 214	256	-
CPU 215/216	256	256

Figure 28

INTERRUPTS				
Instruction	Mnemonic & Operand(s)	Description	STL Status Elements	VALID OPERANDS
Pass Token	PASS	User program has completed its use of the token hold period and is returning control to the system.	STK	

Figure 29

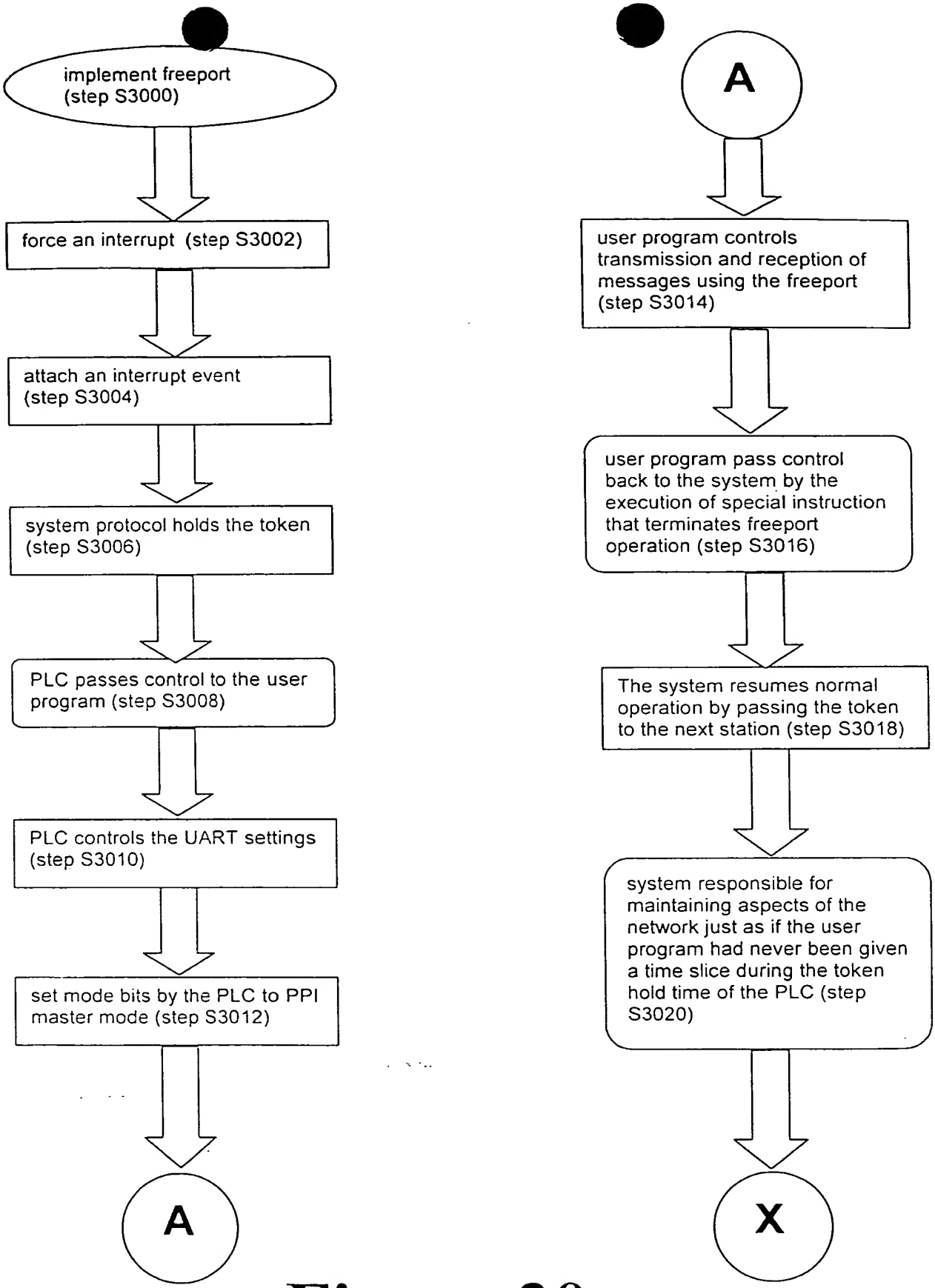

implement freeport (step S3000)
force an interrupt (step S3002)
attach an interrupt event (step S3004)
system protocol holds the token (step S3006)
PLC passes control to the user program (step S3008)
PLC controls the UART settings (step S3010)
set mode bits by the PLC to PPI master mode (step S3012)
A
A
user program controls transmission and reception of messages using the freeport (step S3014)
user program pass control back to the system by the execution of special instruction that terminates freeport operation (step S3016)
The system resumes normal operation by passing the token to the next station (step S3018)
system responsible for maintaining aspects of the network just as if the user program had never been given a time slice during the token hold time of the PLC (step S3020)
X

Figure 30a

SM Bit Definition (Read/Write)

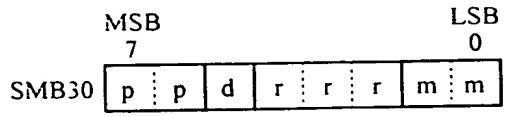
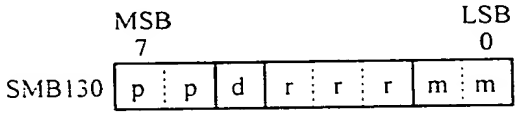
SM Bits	Description
SM30	Port 0: Communication port usage  pp (Parity) '00' - no parity '01' - even parity '10' - no parity '11' - odd parity d (Data bits/char) '0' - 8 bits/char '1' - 7 bits/char rrr (Baud Rate) '000' - 38,400 '001' - 19,200 '010' - 9600 '011' - 4800 '100' - 2400 '101' - 1200 '110' - 600 '111' - 300 mm (Protocol) '00' - PPI Slave (default) '01' - Freeport '10' - PPI Master '11' - reserved (PPI Slave) When the user selects the code mm = 10 (PPI Master), the PLC will become a master on the network allowing the NETR and NETW instructions to be executed. Bits 2 through 7 are ignored in PPI modes. In PPI Master mode with the token acquired interrupt enabled, these bits are used to setup the UART prior to transferring control to the user's program.
SM130	Port 1: Communication port usage (CPU 216 only)  pp (Parity) '00' - no parity '01' - even parity '10' - no parity '11' - odd parity d (Data bits/char) '0' - 8 bits/char '1' - 7 bits/char rrr (Baud Rate) '000' - 38,400 '001' - 19,200 '010' - 9600 '011' - 4800 '100' - 2400 '101' - 1200 '110' - 600 '111' - 300 mm (Protocol) '00' - PPI Slave (default) '01' - Freeport '10' - PPI Master '11' - reserved (PPI Slave) When the user selects the code mm = 10 (PPI Master), the PLC will become a master on the network allowing the NETR and NETW instructions to be executed. Bits 2 through 7 are ignored in PPI modes. In PPI Master mode with the token acquired interrupt enabled, these bits are used to setup the UART prior to transferring control to the user's program.

Figure 30b

SM Bit Definition (Read/Write) (continue	
SM Bits	**Description**
SM87	Port 0: Receive message control byte. (CPU 212/214/216) MSB 7 … LSB 0 en | sc | ec | il | c/m | tmr | bk | 0 en: Enable/disable receive message bit is checked each time the RCV instruction is executed. If this bit is a "0", then the receive message function is disabled. If this bit is a "1", then the receive message function is enabled. sc: 0 - ignore SMB88; 1 - use the value of SMB88 to detect start of message ec: 0 - ignore SMB89; 1 - use the value of SMB89 to detect end of message il: 0 - ignore SMW90; 1 - use the value of SMW90 to detect an idle line condition[1] c/m: 0 - use timer as an inter-character timer; 1 - use timer as a message timer tmr: 0 - ignore SMW92; 1 - terminate receive if the time period in SMW92 is exceeded bk: 0 - ignore break conditions; 1 - use break condition as start of message detection [1]By setting the sc and bk bits to 0 and the en, il, c/m and tmr bits to 1 with an idle line timer value of zero, SMW92 will be used to time out the RCV instruction without receiving any characters. If the timer is not used (tmr = 0), then any character received will be used as start of message. The bits of the message interrupt control byte are used to define the criteria by which the message is identified. Both start of message and end of message criteria are defined. To determine the start of a message either of two sets of logically Anded start of message criteria must be true and must occur in sequence (idle line followed by start character or break followed by start character). To determine the end of a message the enabled end of message criteria are logically ORed. The equations for start and stop criteria are given below: Start of Message = il * sc + bk * sc End of Message = ec + tmr + maximum character count reached Note: Receive will automatically be terminated by an overrun or a parity error (if enabled).
SM88	Port 0: Start of message character. (CPU 212/214/216)
SM89	Port 0: End of message character. (CPU 212/214/216)
SM90 SM91	Port 0: Idle line time period given in milliseconds. The first character received after the idle line time has expired is the start of a new message. SM90 is MSB. (CPU 212/214/216)
SM92 SM93	Port 0: Inter-character/message timer timeout value given in milliseconds. If the time period is exceeded, the receive message is terminated. SM92 is MSB. (CPU 212/214/216)
SM94	Port 0: Maximum number of characters to be received (1 to 255 bytes) (CPU 212/214/216)

Figure 30c

SM Bit Definition (Read/Write) (continue	
SM Bits	**Description**
SM187	Port 1: Message interrupt control byte (CPU 216 only) MSB 7 … LSB 0 en | sc | ec | il | c/m | tmr | bk | 0 en: Enable/disable receive message bit is checked each time the RCV instruction is executed. If this bit is a "0", then the receive message function is disabled. If this bit is a "1", then the receive message function is enabled. sc: 0 - ignore SMB188; 1 - use the value of SMB188 to detect start of message ec: 0 - ignore SMB89; 1 - use the value of SMB189 to detect end of message il: 0 - ignore SMW190; 1 - use the value of SMW190 to detect an idle line condition[1] c/m: 0 - use timer as an inter-character timer; 1 - use timer as a message timer tmr: 0 - ignore SMW192; 1 - terminate receive if the time period in SMW192 is exceeded bk: 0 - ignore break conditions; 1 - use break condition as start of message detection [1]By setting the sc and bk bits to 0 and the en, il, c/m and tmr bits to 1 with an idle line timer value of zero, SMW192 will be used to time out the RCV instruction without receiving any characters. If the timer is not used (tmr = 0), then any character received will be used as start of message. The bits of the message interrupt control byte are used to define the criteria by which the message is identified. Both start of message and end of message criteria are defined. To determine the start of a message either of two sets of logically Anded start of message criteria must be true and must occur in sequence (idle line followed by start character or break followed by start character). To determine the end of a message the enabled end of message criteria are logically Ored. The equations for start and stop criteria are given below: Start of Message = il * sc + bk * sc End of Message = ec + tmr + maximum character count reached Note: Receive will automatically be terminated by an overrun or a parity error (if enabled).
SM188	Port 1: Start of message character (CPU 216 only)
SM189	Port 1: End of message character (CPU 216 only)
SM190 SM191	Port 1: Idle line time period given in milliseconds. The first character received after the idle line time has expired is the start of a new message. SM190 is MSB. (CPU 216 only)
SM192 SM193	Port 1: Inter-character/message timer timeout value given in milliseconds. If the time period is exceeded, the receive message is terminated. SM192 is MSB. (CPU 216 only)
SM194	Port 1: Maximum number of characters to be received (1 to 255 bytes) (CPU 216 only)

Figure 30d

PLC Type	Port 0			Port 1		
	Number of Connections		Buffer Size	Number of Connections		Buffer Size
	Total	Reserved	(Bytes)	Total	Reserved	(Bytes)
CPU 221	4	1 - PG 1 - OP	128/256	-	-	-
CPU 222	4	1 - PG 1 - OP	128/256	-	-	-
CPU 224	4	1 - PG 1 - OP	128/256	-	-	-
CPU 226	4	1 - PG 1 - OP	128/256	4	1 - PG 1 - OP	128/256

Figure 30e

PLC Type	Port 0			Port 1		
	Number of Connections		Buffer Size	Number of Connections		Buffer Size
	Total	Reserved	(Bytes)	Total	Reserved	(Bytes)
CPU 221	4	1 - PG 1 - OP	128/256	-	-	-
CPU 222	4	1 - PG 1 - OP	128/256	-	-	-
CPU 224	4	1 - PG 1 - OP	128/256	-	-	-
CPU 226	4	1 - PG 1 - OP	128/256	4	1 - PG 1 - OP	128/256

Figure 30e

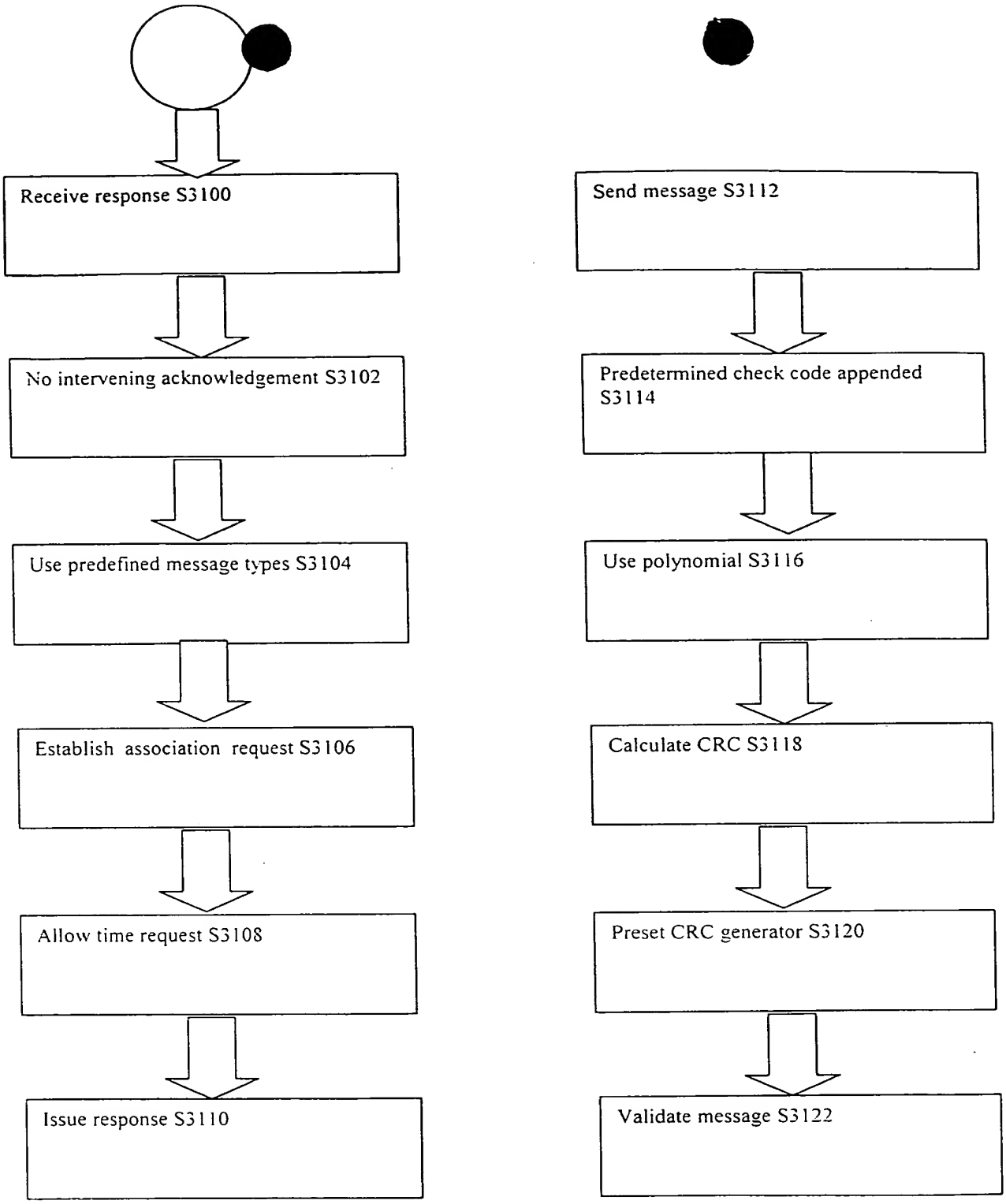

Receive response S3100
No intervening acknowledgement S3102
Use predefined message types S3104
Establish association request S3106
Allow time request S3108
Issue response S3110
Send message S3112
Predetermined check code appended S3114
Use polynomial S3116
Calculate CRC S3118
Preset CRC generator S3120
Validate message S3122

Figure 31

PROGRAM CONTROL FUNCTIONS				
Instruction	Mnemonic & Operand(s)	Description	STL Status Element	VALID OPERANDS
Hide	HIDE n, a, p	The HIDE label marks the start of a block of n instructions that are encrypted using the password, p, when a is non-zero.	STK, SMB1	Enable: None n: KW (2 bytes) (UI) a: KW (2 bytes) (UI) p: KD (4 bytes) (UI)

Figure 32a

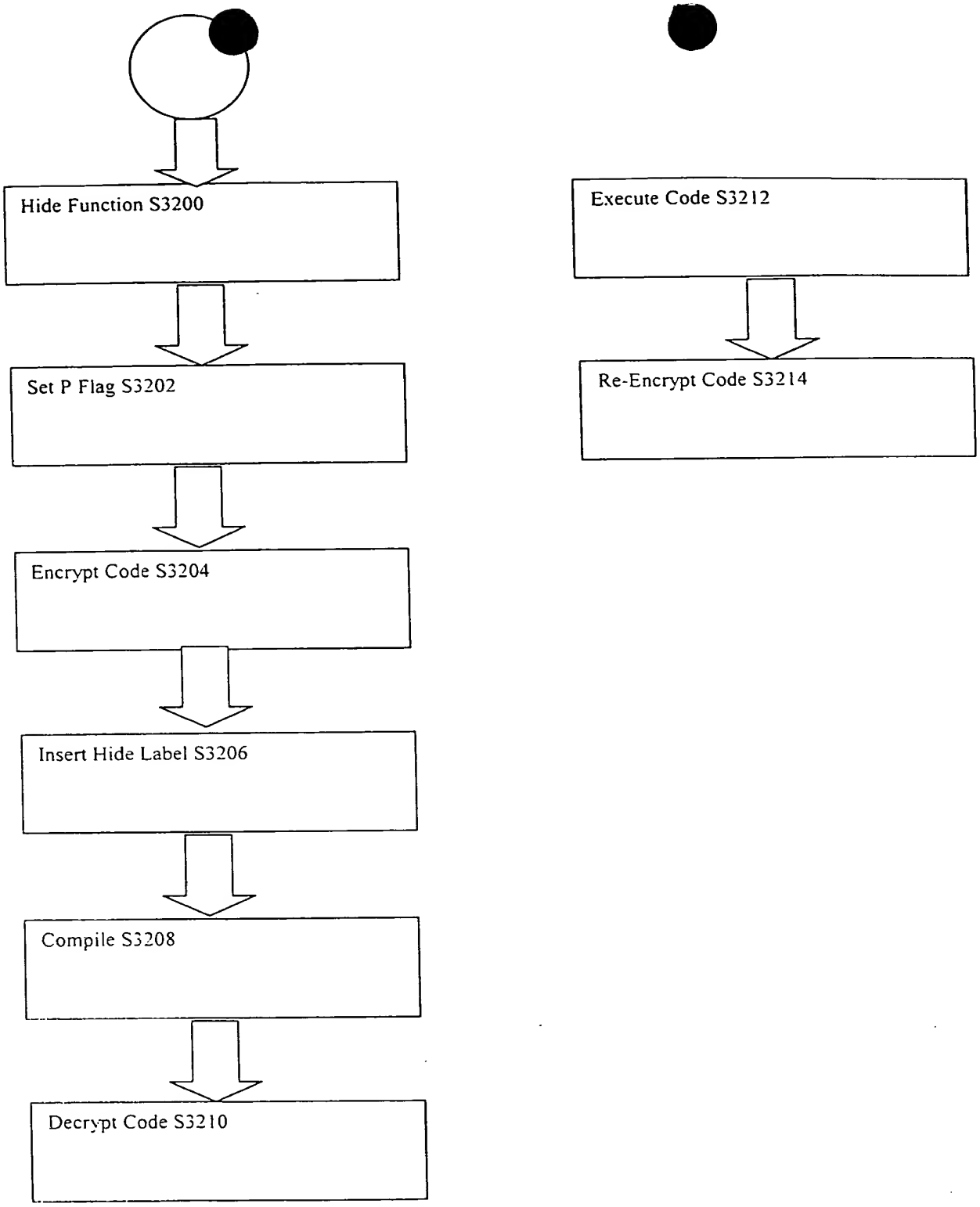

Hide Function S3200
Set P Flag S3202
Encrypt Code S3204
Insert Hide Label S3206
Compile S3208
Decrypt Code S3210
Execute Code S3212
Re-Encrypt Code S3214

Figure 32b

PROGRAM CONTROL FUNCTIONS				
Instruction	Mnemonic & Operand(s)	Description	STL Status Element	VALID OPERANDS
System Function Call	SFC f, s, #_parms, p0, p1, p2, ...	When S0 = 1, execute the system function identified by f and the sub-function identified by s.	STK, SMB1, <p0>, <p1>, <p2>, ...	Enable: S0 f: KW (UI) 0-65536 s: KW (UI) 0-65536 #parms: KB (UI) 0-16 p0, ... : Bit, Byte, Word, Dword Bit V, I, Q, M, SM, S, T, C, L Byte VB, IB, QB, MB, SMB, SB, KB, LB Word VW, T, C, IW, QW, MW, SMW, SW, LW, KW Dword VD, ID, QD, MD, SMD, SD, HC, LD, KD, &VB, &IB, &QB, &MB, &T, &C, &SB Note: Constants and address pointer specifications are not allowed for output or input/output parameters.

Figure 33a

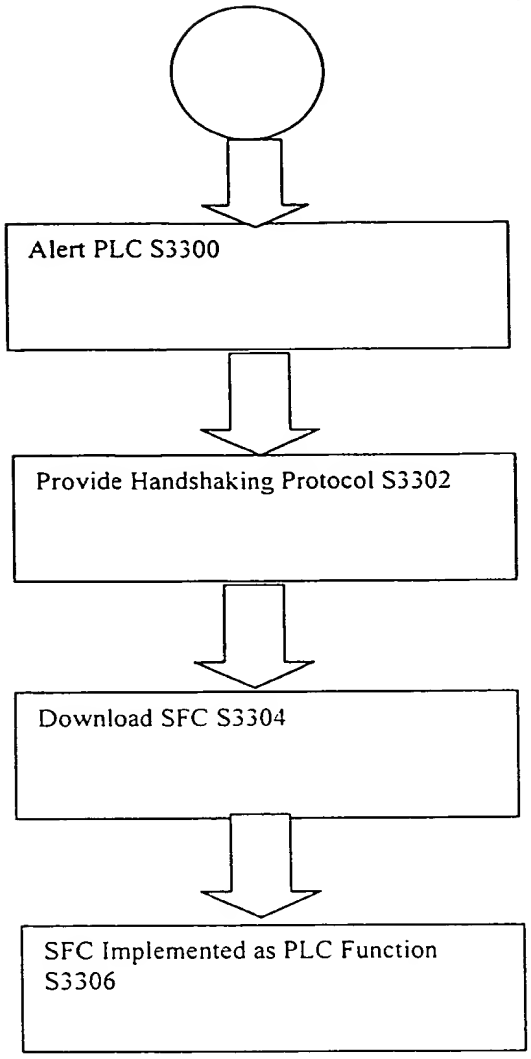

Alert PLC S3300
Provide Handshaking Protocol S3302
Download SFC S3304
SFC Implemented as PLC Function S3306

Figure 33b

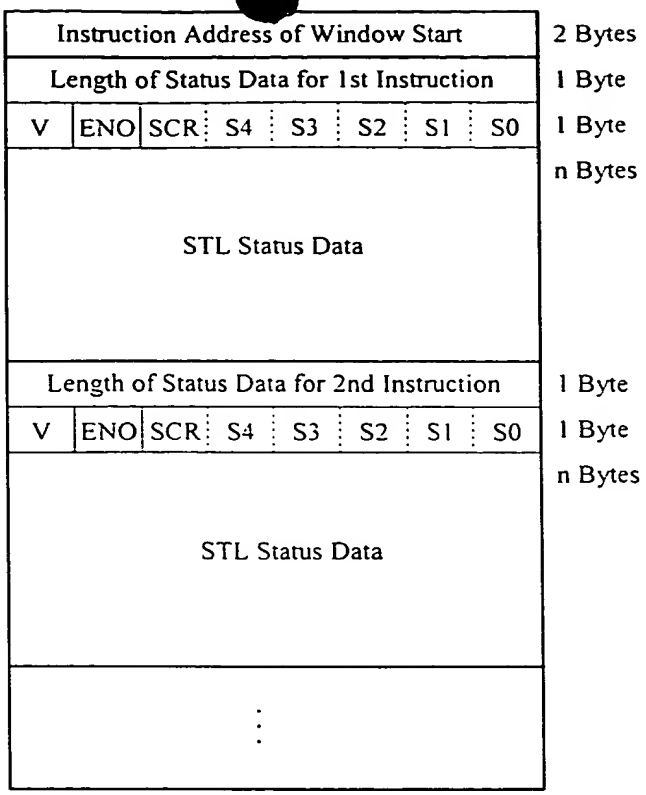


Figure 34a

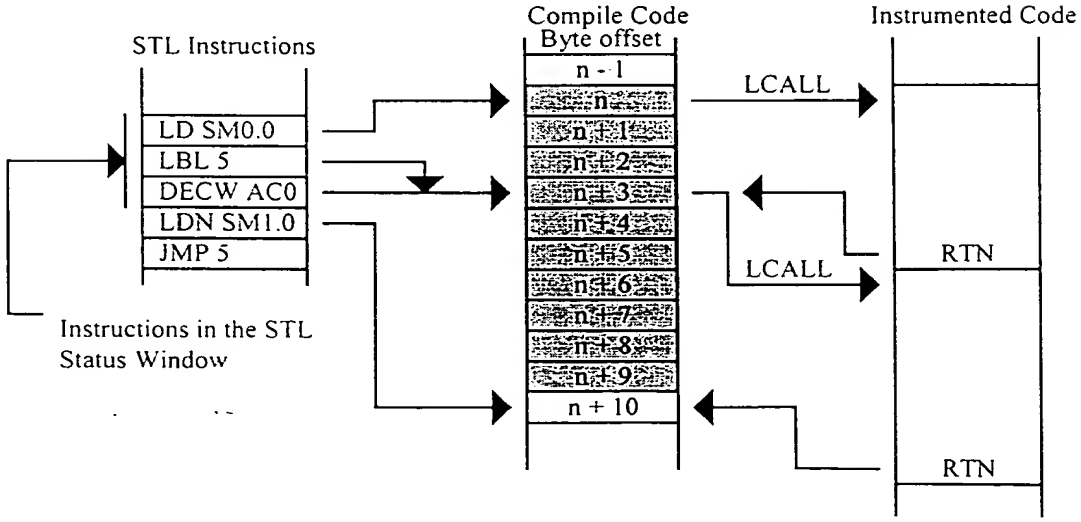


Figure 34b

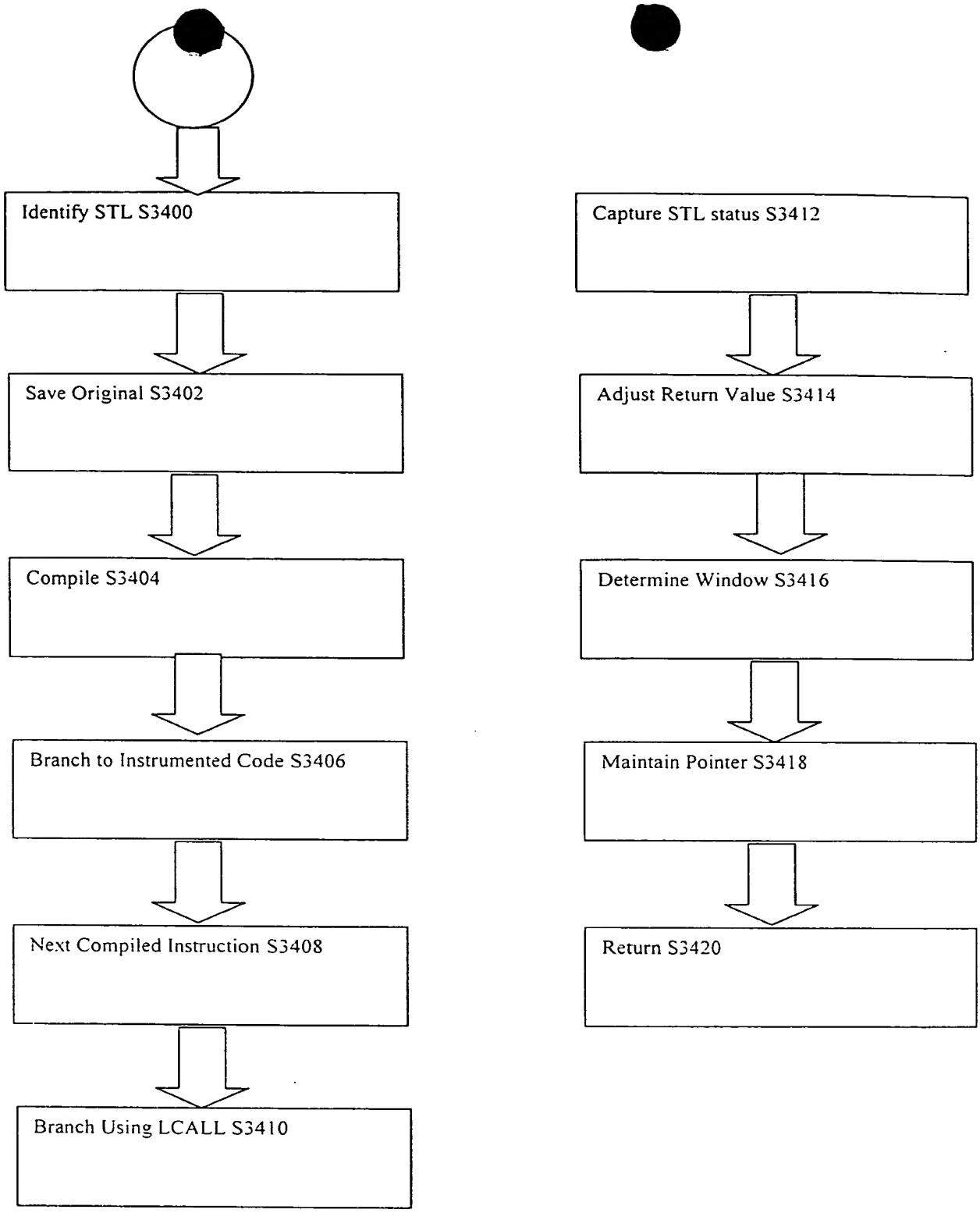

Identify STL S3400
Save Original S3402
Compile S3404
Branch to Instrumented Code S3406
Next Compiled Instruction S3408
Branch Using LCALL S3410
Capture STL status S3412
Adjust Return Value S3414
Determine Window S3416
Maintain Pointer S3418
Return S3420

Figure 34c

	Basic Instr.	Compiled To	Bytes	Cycles
1	LD	RLC A	1	1
		MOV C, <bit>	2	1
2	LDN	RLC A	1	1
		MOV C, <bit>	2	1
		CPL C	1	1
3	AN	ANL C, <bit>	2	2
		NOP	1	1
4	AN	ANL C, /<bit>	2	2
		NOP	1	1
5	O	ORL C, <bit>	2	2
		NOP	1	1
6	ON	ORL C, /<bit>	2	2
		NOP	1	1
7	ST	MOV <bit>, C	2	2
		NOP	1	1
8	ALD	ANL C, ACC.0	2	2
		RR A	1	1
9	OLD	ORL C, ACC.0	2	2
		RR A	1	1
10	NOT	CPL C	1	1
		NOP	1	1
		NOP	1	1
11	LPS	RL A	1	1
		MOV ACC.0, A	2	2
12	LPP	RRC A	1	1
		NOP	1	1
		NOP	1	1
13	LRD	MOV C, ACC.0	2	2
14	RET	RET	1	2
		NOP	1	1
		NOP	1	1
15	INT	CLR A	1	1
		NOP	1	1
		NOP	1	1
16	END	JNC S+0	2	2
		RET	1	2
17	CRETI	JNC S+0	2	2
		RET	1	2

Figure 34d

Housing	Height	Width	Depth
CPU 221	80 mm	90 mm	62 mm
CPU 222	80 mm	90 mm	62 mm
CPU 224	80 mm	120.5 mm	62 mm
CPU 226	80 mm	190 mm	62 mm
8 Point I/O Module	80 mm	46 mm	62 mm
16 Point I/O Module	80 mm	71.2 mm	62 mm
32 Point I/O Module	80 mm	125 mm	62 mm
Intelligent Module	80 mm	90 mm	62 mm

Figure 35a

3900

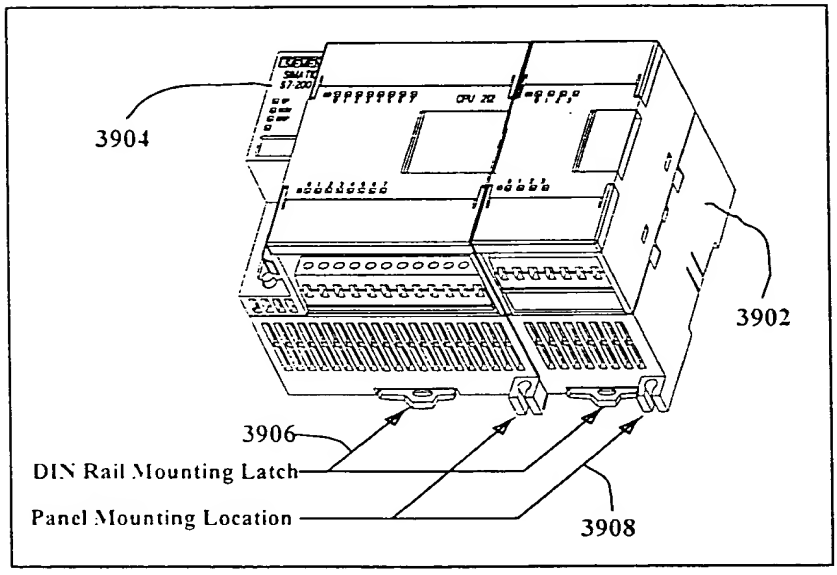


Figure 35b

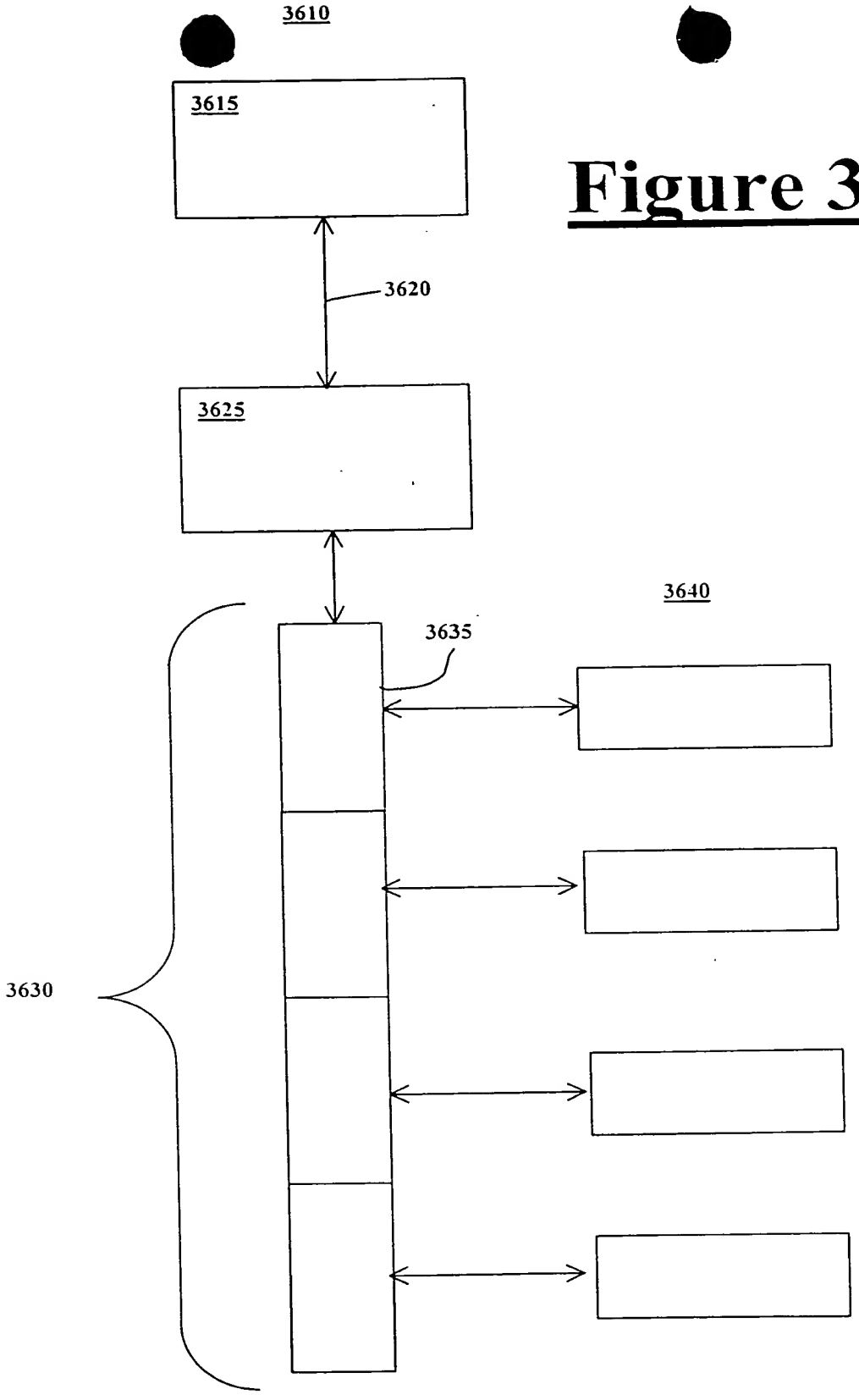


Figure 36

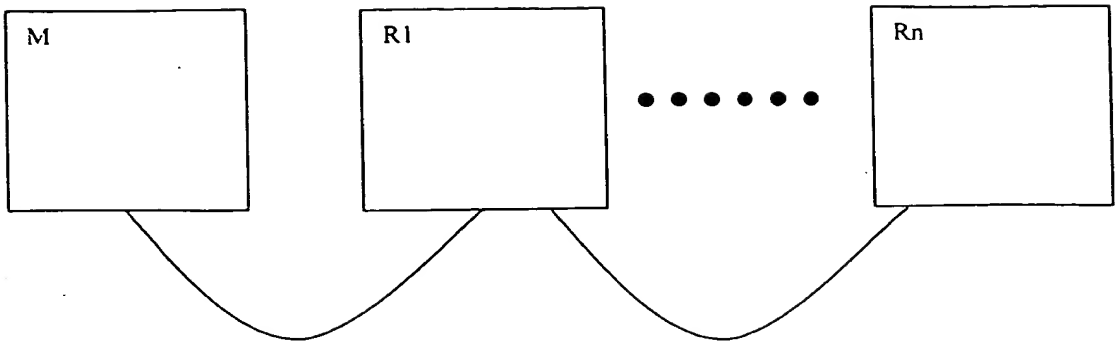


Figure 37a

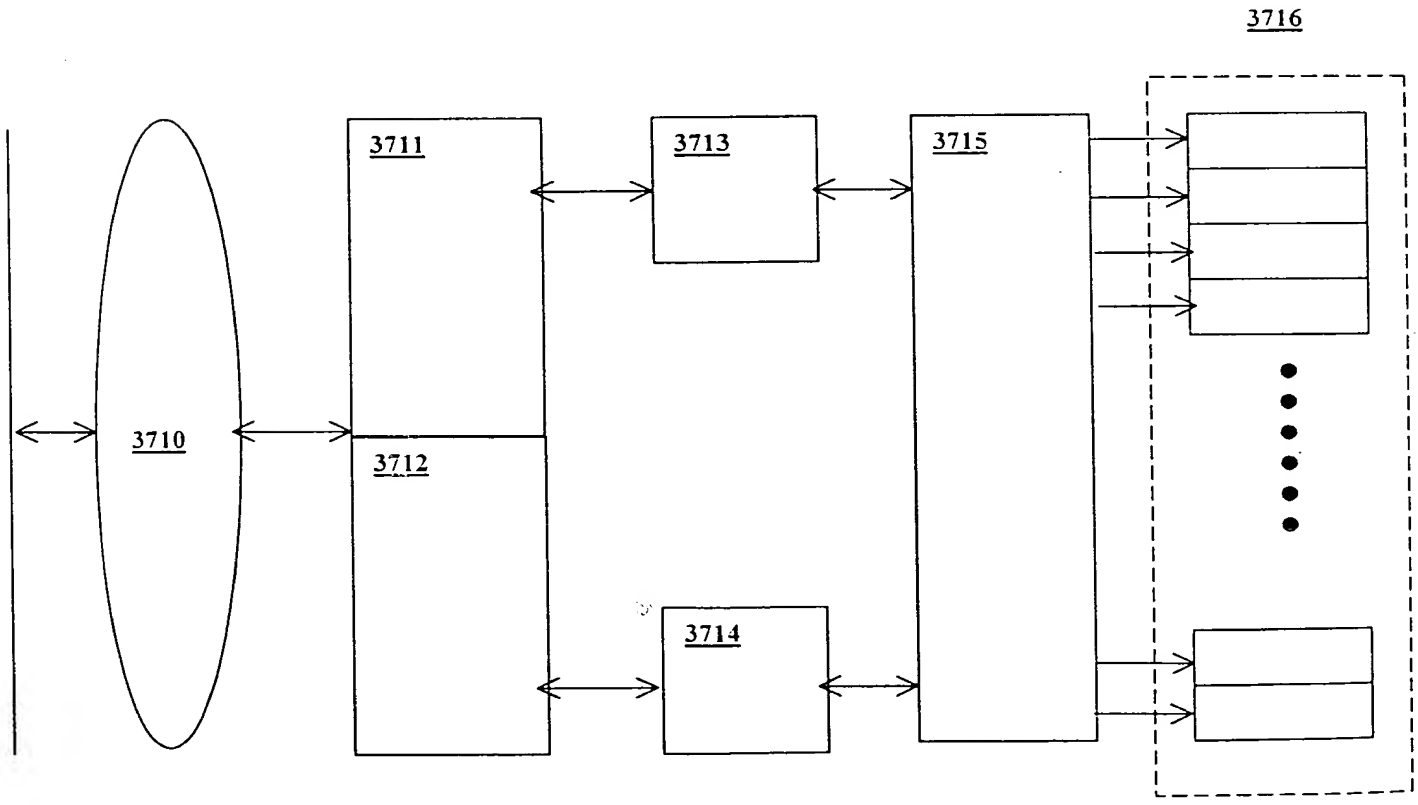


Figure 37b

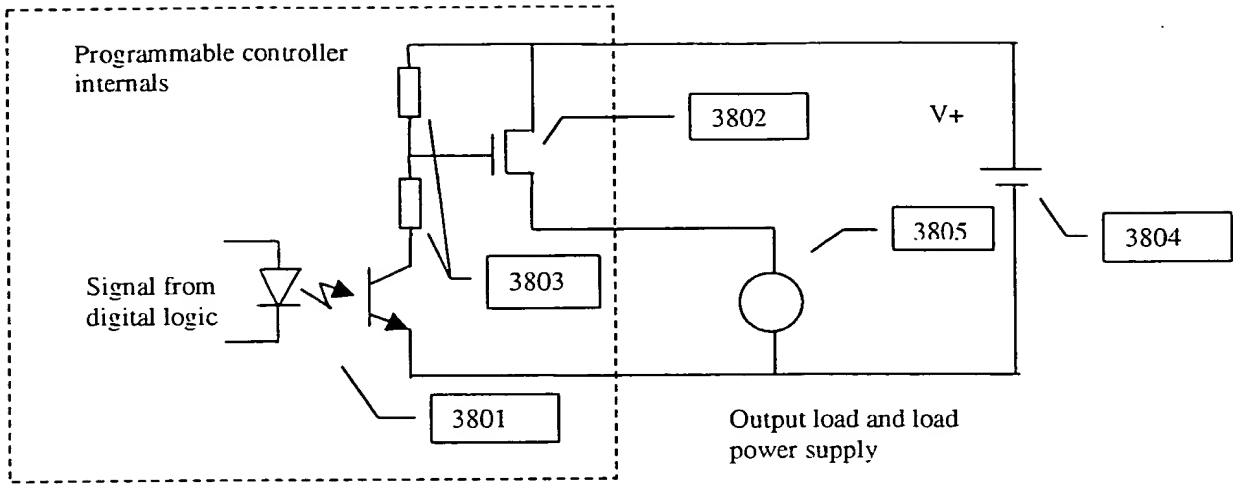


Figure 38 : Typical DC Output of Programmable Logic Controller FET Type Sourcing Output

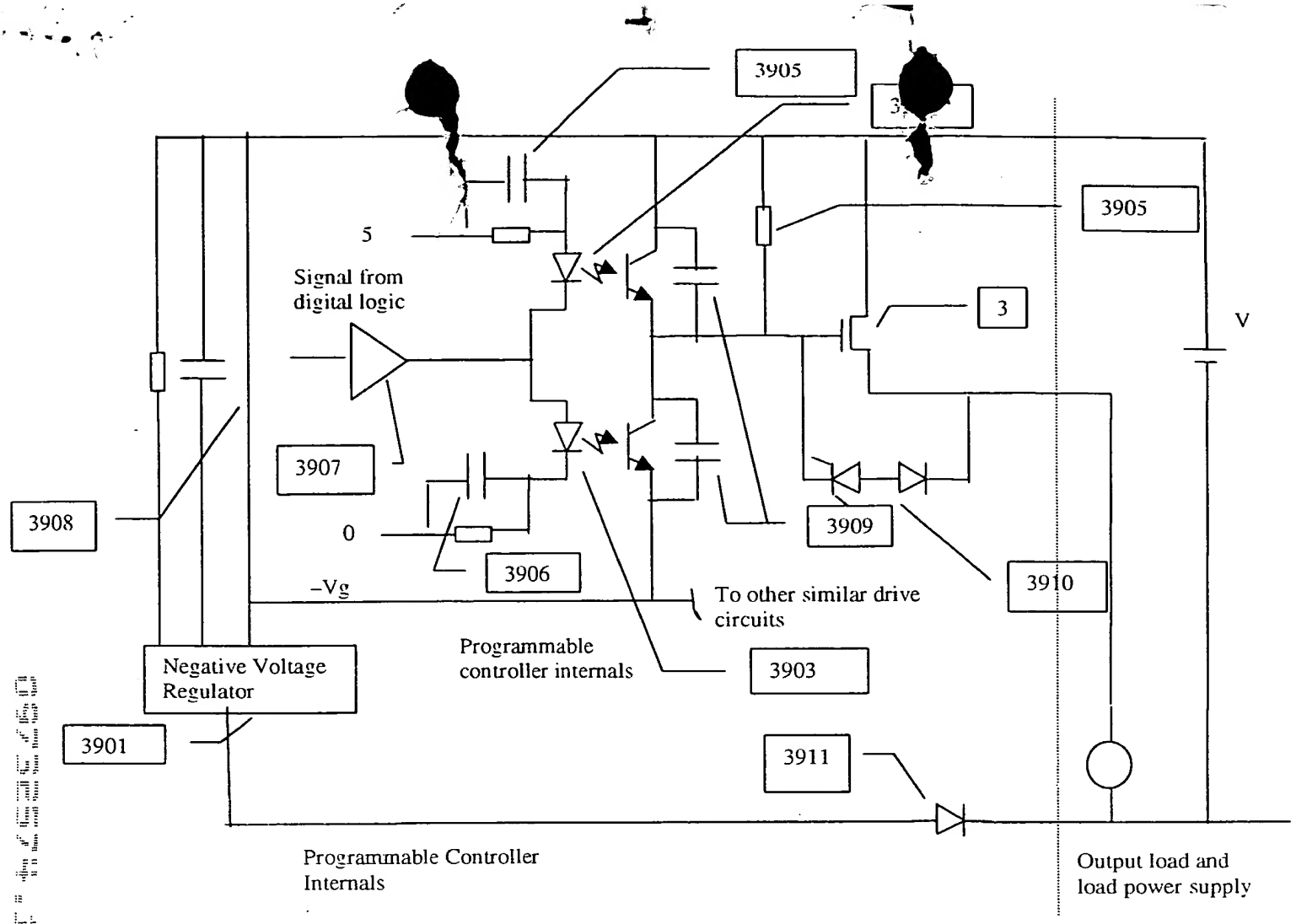


Figure 39 : High speed DC output of Programmable Logic Controller, using push-pull optocoupler circuit.